# Associative Memory Based Hardware Design for An OCR System and Prototyping with FPGA

Ali Ahmadi, M.D.Anwarul Abedin, Kazuhiro Kamimura, Yoshinori Shirakawa, Hans Jürgen Mattausch, and Tetsushi Koide
Research Center for Nanodevices Systems, Hiroshima University, 1-4-2 Kagamiyama, Higashi-Hiroshima, 739-8527, Japan

## 1. Introduction

Much progress has been already made over the past years on development of hardware for optical character recognition (OCR) systems [1] and different movable OCR products are presently in the market [2] but yet they hardly ever afford the desired robustness and hardware size, simultaneously. In this research we propose an associative memory based OCR system for real-time character recognition implemented in an FPGA architecture. The prototype of associative memory we use here as the main classifier is already designed in our lab [3] and has a mixed analog-digital fully-parallel architecture for nearest Hamming/ Manhattan-distance search. In this work, to have a fast prototyping the OCR preprocessing steps as well as the associative memory unit are mapped on the FPGA and the system performance is evaluated with some real data samples.

## 2. OCR Processing Units

The main steps considered for OCR process are: *data reading*, *binarizing*, *noise removal*, *image labeling*, *segmentation*, and *classification*. In data reading step the data of each text line are scanned continuously as a sequence of thin frames by moving a reading device (scanner sensor) on the text. The frames between each two word spaces are collected and form a larger frame as a gray-scale bitmap array which contains all the word characters. In the binarizing step this frame is binarized to a simple black-white bitmap by taking a local threshold value extracted via a mean filter. In order to remove noise from the frame, a median filter with neighborhood of 2×2 is applied. Next, by employing a sequential labeling algorithm different segments of the image frame are labeled and segments larger than a threshold level are recognized as a single character. Further detail about labeling algorithm is given in the next Section.

To have an accurate classification the size of each character is normalized to 16×16 pixels before classification. We use a bilinear interpolation algorithm for resizing the character bitmap. The last and main step of the process is character classification which is carried out by a nearest-distance search algorithm applying the associative memory. The normalized segmented character is matched as a 256 bits vector to a number of reference patterns using the Manhattan distance measure and the reference pattern with minimum distance is considered as the winner class.

Figure 1 shows a simple schematic of the associative memory used as the main classifier of the system. A number of k-bit digital subtraction and absolute-value calculation units compare the W binaries in all rows of the memory field with the reference data in parallel. Low power dissipation of the system is achieved by an individual power regulation from the signal-regulation units for each input-signal source. The transistor-count is only 6 per row. More detailed information about the associative memory performance can be found in [3]. Table 1 shows the characteristic performance data of designed associative memory depending on the Hamming and Manhattan distance measure.

## 3. System Prototyping with FPGA

Major blocks of preprocessing steps are designed and implemented in an FPGA architecture. We use the Altera Stratix family DSP development kit as the main platform. Figure 2 shows the schematic of the binarizing block. As is explained in Section 2, a mean filter is used for determining the local threshold values. Each image frame is read as a sequence of pixels and as can be seen in Fig. 2, eight registers (DFF and FIFO), preparing different delay times, are used to provide the 8 neighborhood points for each pixel. The binarizing process is implemented using 13 operators including 8 registers, a parallel-adder, divider, and comparator.

The hardware design used for mean filter in binarizing block can be generalized to all finite impulse response (FIR) filters and so is applied in noise removing block (as a median filter) and labeling block, as well. In the labeling process, we keep the labels of preceding neighbor pixels (4 pixels' labels) and decide if the current pixel belongs to one of the preceding labels or gets a new label. The labels of all image pixels are then extracted and written in the label memory in this way. In case of facing with equivalence labels (like in character V) the label equivalences are recorded in a local table (two buffers SLB1 and SLB2). Once the labeling process is terminated, the image memory is scanned once again for segmentation task. Figure 3 depicts the algorithm used for segmentation. We scan the label memory N times each time searching label $L_i$ (i=1…N). The label read from memory is then searched within the SLB1 buffer and if found, will be replaced with the equivalent label from SLB2. The addresses of pixels with label $L_i$ are written in a new memory (SM) as a distinct segment $L_i$. Next, the boundaries of segment $L_i$ are identified and a new segment vector with binary values 0 and 1 is generated for this boundary based on the addresses saved in SM memory.

As for modeling the fully-parallel functionality of the associative memory in the FPGA, four dual-port SRAM memory blocks each containing 32 data words are applied. Taking this structure and a memory I/O bus of 256 bits width, we can have 16 parallel matchings within one clock cycle. Figure 4 shows a simple schematic of the design.

The design is described in Verilog-HDL and synthesized using the Synplify-pro compiler and then implemented in the Altera FPGA family using the QuartusII tool for placement and routing. Using Stratix DSP development kit EP1S80 and a clock frequency of 50 MHz, a total number of 2,337 logic cells and 36 Kbits of SRAM memory are used for placement and routing of associative memory part. The timing simulation results are reported in the next Section.

## 4. Analysis of Simulation Results

We examined the system with some real data samples of English text characters and evaluated the results with a software program. A total number of 16 data set including different fonts (Times and Arial), noisy data, color background data, slightly rotated data, and data with different resolution were gathered and tested. Each set contained 26 characters and as mentioned before, each data sample is considered as a 256 bits vector. The experimental results of distance-matching between data vectors and reference patterns showed overall of 2 misclassification cases (0.5%) for noisy data and zero case for other data type. The minimum distance between winner and nearest-loser over all the data samples is averagely 2.5 which is not yet reliable enough. The minimum distance of winner and second loser is 13 bits.

Figure 5(a) indicates the winner-input distance for different data samples. The average winner-input distance for all the input samples was calculated and found as 31 bits. Having this distance and referring to plot of Fig. 5(b) which gives the typical winner search time of the associative memory according to winner-input distance, we can find the average search time of 128 ns for classification of each test sample. This is the search time within 128 reference patterns and will be changed in case of increase in reference patterns number. Comparing to other OCR products existing in the market, however this prototype model is not still robust enough but is advantageous in terms of classification time and hardware size. We are also planning to develop the system with a learning algorithm for optimizing the reference patterns selection process.

## References

[1] http://www.ocr.researchcccc.com/directory/online-ocr.html

[2] For example Wizcom *Quicktionary* a mobile scanner dictionary, and *IrisPen* a handheld scanner pen.

[3] Y.Yano, T. Koide, H.J. Mattausch, Associative Memory with Fully Parallel Nearest-Manhattan-distance Search for Low-power Real-time Single-chip Applications, Proc. of ASP-DAC'2004, pp. 543 – 544, Japan, 2004.

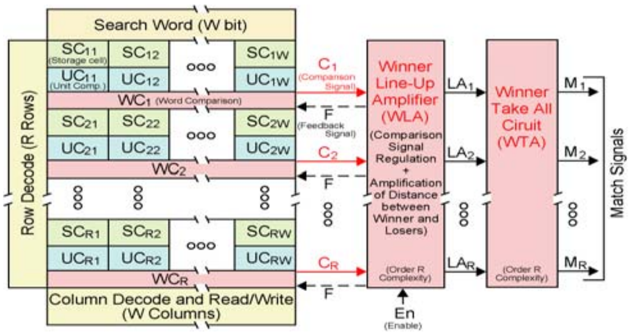
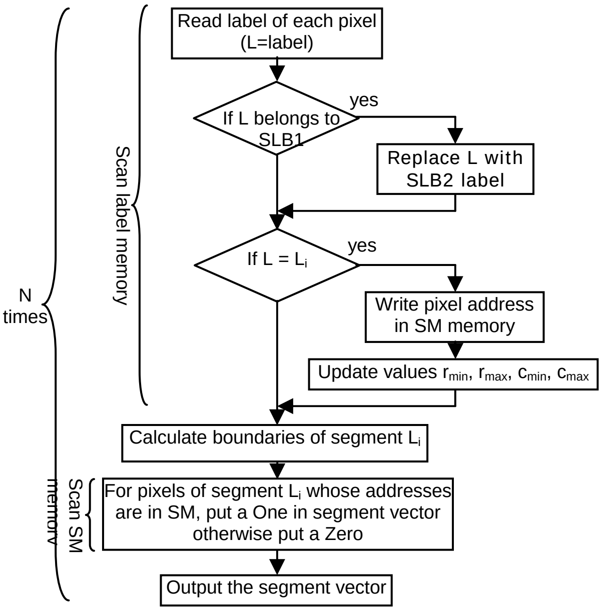


Fig. 1. Associative memory architecture.

Table 1: Characteristic data of designed associative memory test chip.

| Distance Measure | Hamming | Manhattan (5 bit) |
|---|---|---|
| Memory Field | 32 x 768 | 128 x 80 |
| Technology | 0.6 µm CMOS | 0.35 µm CMOS |
| Area | 9.11 mm$^2$ | 8.6 mm$^2$ |
| Search Range | 0 - 400 bit | 0 - 480 bit |
| Winner-Search Time | < 70 nsec | < 190 nsec |
| Performance | 1.34 TOPS | 160 GOPS |
| Power Dissipation | 43 mW | 91 mW |
| Supply Voltage | 3.3V | 3.3V |



Read label of each pixel (L=label)
If L belongs to SLB1
yes
Replace L with SLB2 label
If L = Li
yes
Write pixel address in SM memory
Update values rmin, rmax, cmin, cmax
Calculate boundaries of segment Li
For pixels of segment Li whose addresses are in SM, put a One in segment vector otherwise put a Zero
Output the segment vector
Scan label memory
Scan SM memory
N times

Fig. 3. Flowchart of segmentation procedure. N is number of distinct labels (Li , i=1…N), SLB1, SLB2 are buffers for recording equivalent labels, SM is a temporary memory for saving addresses of pixels with label Li, and $c_{min}$, $c_{max}$, $r_{min}$, $r_{max}$ are min. row, max. row, min. column, max. column of segment $L_i$, respectively.

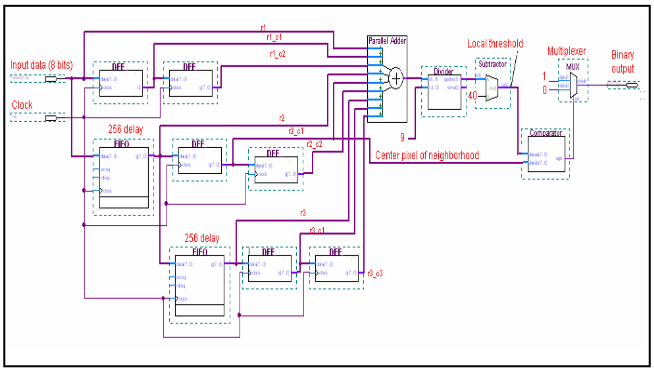


Fig. 2. The schematic of the binarizing block using local threshold algorithm implemented in an FPGA architecture.

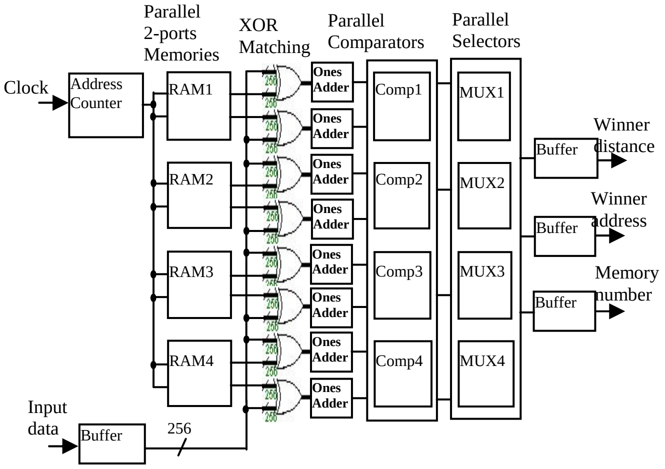


Fig. 4. The schematic of associative memory mapped with 4 dual-port RAMs in an FPGA architecture. 16 matchings are performed within each clock cycle.

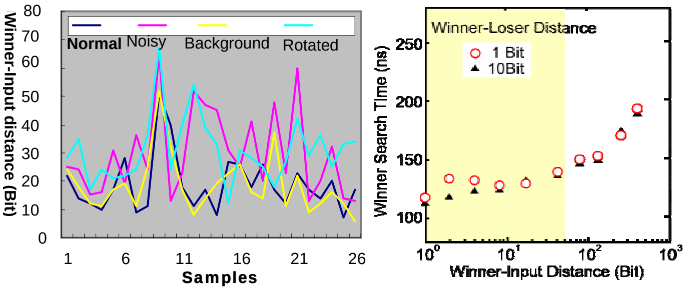


Fig. 5 (a). Winner-Input distance for different data of font Times. (b) Average winner search times in associative memory.

Hiroshima University
Research Center for Nanodevices and Systems (RCNS)
System Design and Architecture Research Division

# Associative Memory Based Hardware Design for An OCR System and Prototyping with FPGA

A. Ahmadi, M.A. Abedin, K. Kamimura, Y. Shirakawa, H.J. Mattausch, and T. Koide
Research Center for Nanodevices and Systems, Hiroshima University

## Background & Research Objective

**Optical character recognition (OCR) systems have been widely used in recent years and various approaches are applied for developing their hardware and processing algorithms.**

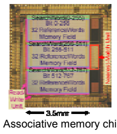
**As for a small mobile OCR, ex. a cognitive pen, usually an ideal model is thought as a system with high accuracy and speed, and minimum hardware size & power dissipation at the same time.**

**We propose an associative memory based OCR system for real-time character recognition implemented in an FPGA architecture.**

**An associative memory-based OCR with qualifications:**

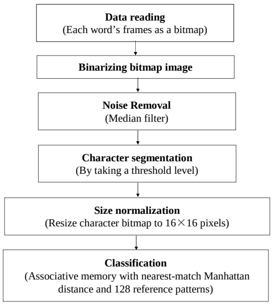
**1- Recognition of printed characters & words**
**2- Robustness to Noise, Rotation, Color**
**3- Applicable to different Fonts**
**4- High Speed**

**5- Learning capability**



Associative memory chip

## Processing Steps



Data reading
(Each word's frames as a bitmap)

Binarizing bitmap image

Noise Removal
(Median filter)

Character segmentation
(By taking a threshold level)

Size normalization
(Resize character bitmap to 16×16 pixels)

Classification
(Associative memory with nearest-match Manhattan distance and 128 reference patterns)

Block diagram of system steps.

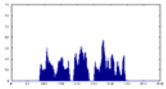
### Data Reading

**Real-time data reading problems:**

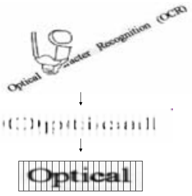
-Rotation & shift

-High rate of noise

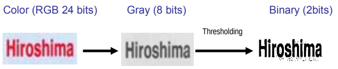
-High speed of recognition needed



A Vertical Projection Profile (VPP) of the characters is used to find the space between word.

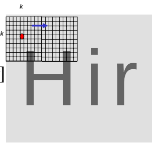


### Binarizing

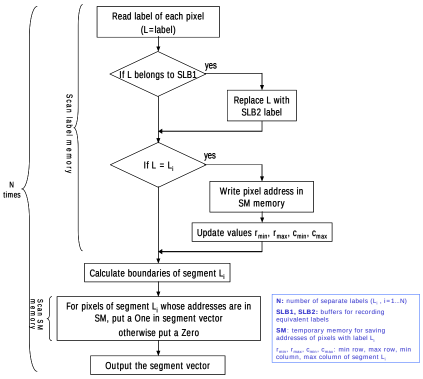


Local thresholding

$$Th_i = \frac{1}{nb}\sum I_j$$
$$j \in [i-k, i+k]$$
$$nb = k \times k$$



For each pixel a neighborhood of size *k* is taken and a local threshold is defined by mean of pixels.



Read label of each pixel (L=label)

If L belongs to SLB1 — yes → Replace L with SLB2 label

If L = $L_i$ — yes → Write pixel address in SM memory → Update values $r_{min}$, $r_{max}$, $c_{min}$, $c_{max}$

Calculate boundaries of segment $L_i$

For pixels of segment $L_i$ whose addresses are in SM, put a One in segment vector otherwise put a Zero

Output the segment vector

**N:** number of separate labels ($L_i$ , i=1...N)

**SLB1, SLB2:** buffers for recording equivalent labels

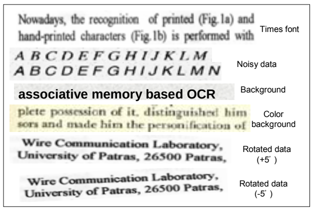
**SM**: temporary memory for saving addresses of pixels with label $L_i$

$r_{min}$, $r_{max}$, $c_{min}$, $c_{max}$: min row, max row, min column, max column of segment $L_i$

Flowchart of data segmentation (after labeling).

## Experimental Results

**The system performance and search time were evaluated by using some real data samples. A total number of 16 data set including different fonts (Times and Arial) and different data type and resolution was used.**

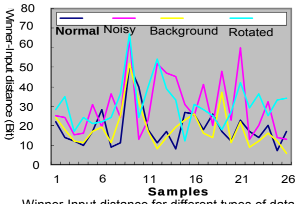


Nowadays, the recognition of printed (Fig.1a) and hand-printed characters (Fig.1b) is performed with
A B C D E F G H I J K L M
A B C D E F G H I J K L M N
associative memory based OCR
plete possession of it, distinguished him
sors and made him the personification of
Wire Communication Laboratory,
University of Patras, 26500 Patras,
Wire Communication Laboratory,
University of Patras, 26500 Patras,

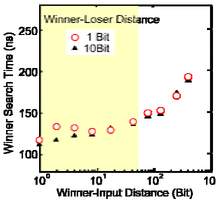
Some text samples used as input data.

Classification results for different data type font: Times, Small characters
(nl: Nearest l321oser)

| Sample Type | dpi | Char. no. | Correct classification no. | Mis-classification no. | Classification to other fonts | Min. dist. of winner & nl | Min. dist. of winner & 2nd loser | Avg. dist. of winner & input | Misclassified cases |
|---|---|---|---|---|---|---|---|---|---|
| Normal | 200 | 26 | 26 | 0 | 0 | 1 | 7 | 28 | |
| Noisy | 200 | 26 | 25 | 1 | 1 | 1 | 10 | 32.9 | g(A)→g(T), i→l |
| Backgrnd | 200 | 26 | 25 | 1 | 0 | 2 | 15 | 35.8 | i(T)→j(A) |
| Rotated | 200 | 26 | 26 | 0 | 1 | 3 | 7 | 44.1 | i(A)→i(T) |
| Normal | 300 | 26 | 26 | 0 | 0 | 5 | 14 | 21 | |
| Noisy | 300 | 26 | 26 | 0 | 0 | 2 | 10 | 26 | |
| Backgrnd | 300 | 26 | 26 | 0 | 0 | 5 | 19 | 25.1 | |
| Rotated | 300 | 26 | 26 | 1 | 1 | 1 | 12 | 40.1 | h→b,<br>i(A)→i(T) |
| Total | | 208 | 206 | 2 | 3 | 2.5 | 13 | 31.6 | |



Winner-Input distance for different types of data for font Times



Winner search times in associative memory

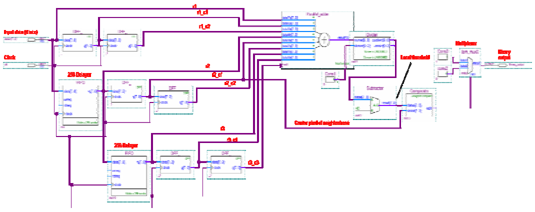
**Average winner-input distance for test samples: 32 bits**

**Average search time: 129 ns**

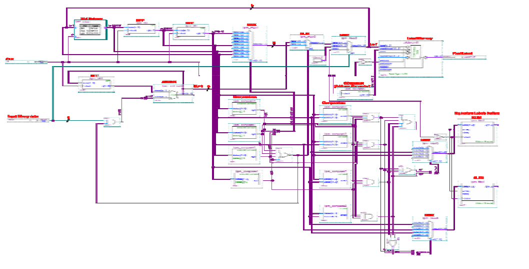
Which is extremely faster than other existing OCR systems.

## Hardware Implementation

**Preprocessing steps are designed and implemented in an FPGA architecture. We use Altera Stratix DSP development kit EP1S80 with a clock of 50 MHz as the main platform.**

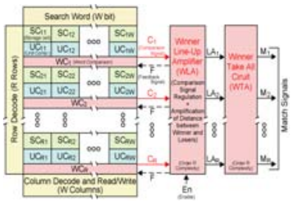


The schematic of binarizing block using local threshold algorithm implemented in an FPGA architecture. The core part of algorithm can be generalized for implementation of any FIR filter.



The schematic of labeling block. Using Stratix DSP development kit EP1S80 a total number of 76 logic cells and 328 Kbits of SRAM memory are used for placement and routing.

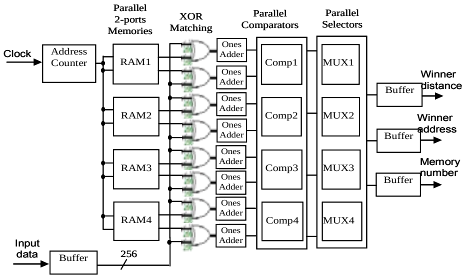
The prototype of associative memory we use here as the main classifier is already designed in our lab as an LSI chip and has a mixed analog-digital fully-parallel architecture for nearest Hamming/ Manhattan-distance search


Search Word (W bit)
Row Decode (R Rows)
SC11
SC12
SC1W
UC11
UC12
UC1W
WC1
SC21
SC22
SC2W
UC21
UC22
UC2W
WC2
SCR1
SCR2
SCRW
UCR1
UCR2
UCRW
WCR
Column Decode and Read/Write (W Columns)
C1
C2
CR
F
Winner Line-Up Amplifier (WLA)
(Comparison Signal Regulation + Amplification of Distance between Winner and Losers)
LA1
LA2
LAR
Winner Take All Circuit (WTA)
M1
M2
MR
Match Signals
En

Architecture of associative memory used as the main classifier of the system.

Characteristic specifications of associative memory designed as an LSI chip

| Distance Measure | Hamming | Manhattan (5 bit) |
|---|---|---|
| Memory Field | 32 x 768 | 128 x 80 |
| Technology | 0.6 μm CMOS | 0.35 μm CMOS |
| Area | 9.11 mm² | 8.6 mm² |
| Search Range | 0 - 400 bit | 0 - 480 bit |
| Winner-Search Time | < 70 nsec | < 190 nsec |
| Performance | 1.34 TOPS | 160 GOPS |
| Power Dissipation | 43 mW | 91 mW |
| Supply Voltage | 3.3V | 3.3V |



The schematic of associative memory mapped with 4 dual-port RAMs in an FPGA architecture. 16 matchings are performed within each clock cycle.

## Conclusions

**1- The proposed associative memory based OCR is advantageous in terms of classification speed and hardware size.**

**2- Due to fully parallel pattern-matching used in the associative memory the average search time for each character is obtained as 129 ns which is very much faster than other existing OCRs.**

**3- We are planning to equip the system with a learning algorithm which updates and optimizes reference patterns continuously over the time using two types of short term and long term memory.**