논문 2007-44SD-9-6

임베디드 시스템 적용을 위한 얼굴검출 하드웨어 설계

(Face detect hardware implementation for embedded system)

김 윤 구*, 정 용 진**

(Yoon Gu Kim and Yong Jin Jeong)

요 약

제한적인 자원을 갖는 임베디드 시스템을 위한 영상처리 하드웨어 설계 시 메모리의 효율적인 구성은 필수적으로 고려할 사항이다. 특히 필터를 이용한 얼굴 검출 하드웨어는 필터와 입력영상을 저장하기 위해 많은 양의 메모리가 소요되기 때문에 효율적인 메모리 구성이 필요하다. 따라서 본 논문은 일반적인 필터방식의 알고리즘을 하드웨어 설계에 적절하도록 보완하여 하드웨어로 설계하였다. 설계된 하드웨어는 알고리즘 특성에 맞추어 적은 양의 내부 메모리를 사용하면서 한번 외부 메모리로부터 읽은 데이터를 다시 읽지 않도록 구성하였고, 데이터 양이 많아 외부 메모리에 저장되어 있는 필터를 효율적으로 사용하기 위해 필터의 일부를 내부 메모리로 복사하는 구조로 설계하였다. 또한 빠른 연산을 위해 여러 클럭이 소모되는 데이터 패스를 파이프라인 구조를 적용하여 연속적으로 메모리 데이터를 읽을 수 있는 구조로 설계하였다. 본 하드웨어는 xilinx 및 ARM 기반의 FPGA 환경에서 검증한 결과 1초에 25 프레임 처리가 가능하며 40KB의 내부 메모리를 사용하였고 삼성 0.18um공정을 이용하여 칩으로 제작 중이다.

Abstract

For image processing hardware, including a face detecting engine, efficient constitution of external and internal memories is a consequential point because huge memory is required to store various signal processing filters and incoming images. In this paper, we modified a face detect algorithm of a general filter method for efficient hardware design. In the hardware, several memory design techniques are presented for efficient handling of image data : re-accessing avoidance with minimized internal memory usage, residing frequently accessed memory and sequence memory accessing. The hardware which can process 25 frame image data per one second with 40KB internal memory was verified by using ARM(S3C2440A) and Virtex4 FPGA and it is being fabricated as a ASIC chip using Samsung CMOS 0.18um technology.

Keywords : 얼굴인식, 얼굴검출, 임베디드 시스템

I. 서 론

얼굴검출은 2단계(얼굴검출, 얼굴인증)로 구성된 얼굴인식 과정의 주요 알고리즘으로써, 입력된 영상에서 얼굴영역을 검출하는 역할을 한다. 얼굴검출에서 검출한 얼굴영역의 정확도에 따라 얼굴인증의 결과에 큰 영향을 미치기 때문에 얼굴검출은 얼굴인식 시스템에서 중요한 부분을 차지한다. 본 논문에서는 이러한 얼굴검출을 임베디드 환경에 적용할 수 있도록 기존의 개발된 알고리즘을 하드웨어로 설계한다. 그러나 하드웨어 설계를 위한 대부분의 얼굴 검출 알고리즘의 문제는 밝기변화에 의해 큰 성능 차이를 보이는 것이다. 이를 해결하기 위해 본 논문은 입력영상을 밝기변화에 관계없는 구조정보로만 구성된 영상으로 변환하는 ICT(Improved Census Transform) 알고리즘을 얼굴검출 알고리즘의 전처리로 사용하여 밝기변화에 안정적인 입력영상을 얻는다. 사용된 ICT 알고리즘은 MCT(Modified Census Transform) 알고리즘을 근간으로 하고 있으나 임베디드 시스템 적용을 위해 개선된 알고리즘이다.

* 학생회원, ** 정회원, 광운대학교 전자통신 공학과
(Dept, of Electronics and Communication Engineering, Kwangwoon University)

※ 본 연구는 교육인적자원부에서 지원받은 2006년 광운대학교 대학특성화사업(차세대 신성장 동력산업을 위한 실감 IT 전문인력 양성사업)의 일환으로 진행되었으며 이에 감사를 드립니다.
접수일자: 2007년4월17일, 수정완료일: 2007년8월17일

II. 알고리즘

기존 얼굴검출 방식은 피부색을 이용하여 검출하는

방식[3], 얼굴의 패턴을 인식하는 방식[4], Haar-Like 방식의 얼굴 피처를 이용하는 방식[5] 등이 사용되었다. 그러나 이와 같은 검출방법은 근본적으로 입력된 영상이 밝기에 따라 픽셀값이 변화되기 때문에 검출 정확도의 한계를 보이고 있다. 따라서 여러 어플리케이션에 적용되기 위한 얼굴 검출을 위해 입력영상을 밝기변화에도 변화가 없는 구조정보만을 포함한 영상으로의 변환이 필요하다. 본 논문에서는 영상 변환 알고리즘인 ICT (Improved Census Transform) 알고리즘을 소개하고 이를 토대로 한 얼굴 검출 알고리즘을 소개한다.

1. ICT(Improved Census Transform) 알고리즘

$$\Gamma_i(X) = a_i\times 2^8 + b_i\times 2^7 + c_i\times 2^6 + d_i\times 2^5 + e_i\times 2^4 + f_i\times 2^3 + g_i\times 2^2 + h_i\times 2^1 + i_i\times 2^0 \quad (1)$$

Bernhard Fröba [1]는 밝기에 따른 입력영상의 변화를 완충해 주기 위해 MCT(Modified Census Transform) 알고리즘을 제안하였다. MCT 알고리즘은 이미지의 픽셀값을 주위 픽셀과의 관계값으로 변환하는 알고리즘으로 그 결과값(커널 인덱스)이 밝기 변화에 비교적 안정적이다. 그러나 MCT 알고리즘의 결과값은 0부터 511의 512개의 경우로 표현되기 때문에 이를 통한 얼굴 검출 알고리즘은 임베디드 시스템 적용에 용의하지 않다. 이는 수많은 체크점으로 구성된 얼굴 필터에서 512 경우의 결과값은 한 체크 점당 512개의 필터값으로 구성되는 것을 의미하므로 필터를 저장하기 위해 많은 양의 메모리를 필요로 하기 때문이다. 따라서 본 논문에서 사용한 얼굴 검출 알고리즘은 그림 2와 같이 MCT 알고리즘과 유사하지만, 하드웨어 설계에 적합하도록 알고리즘을 개선한 ICT 알고리즘을 이용한다.

Pi-4	Pi-3	Pi-2
Pi-1	Pi	Pi+1
Pi+2	Pi+3	Pi+4

$$Z_i = \frac{\sum_{m=-4}^{+4} P_{i+m}}{9} \qquad a_i = \begin{cases} 0 & \text{if } P_{i-4} \leqq Z_i \\ 1 & \text{if } P_{i-4} > Z_i \end{cases}$$

$$b_i = \begin{cases} 0 & \text{if } P_{i-3} \leqq Z_i \\ 1 & \text{if } P_{i-3} > Z_i \end{cases} \quad c_i = \begin{cases} 0 & \text{if } P_{i-2} \leqq Z_i \\ 1 & \text{if } P_{i-2} > Z_i \end{cases}$$

$$d_i = \begin{cases} 0 & \text{if } P_{i-1} \leqq Z_i \\ 1 & \text{if } P_{i-1} > Z_i \end{cases} \quad e_i = \begin{cases} 0 & \text{if } P_i \leqq Z_i \\ 1 & \text{if } P_i > Z_i \end{cases} \quad f_i = \begin{cases} 0 & \text{if } P_{i+1} \leqq Z_i \\ 1 & \text{if } P_{i+1} > Z_i \end{cases}$$

$$g_i = \begin{cases} 0 & \text{if } P_{i+2} \leqq Z_i \\ 1 & \text{if } P_{i+2} > Z_i \end{cases} \quad h_i = \begin{cases} 0 & \text{if } P_{i+3} \leqq Z_i \\ 1 & \text{if } P_{i+3} > Z_i \end{cases} \quad i_i = \begin{cases} 0 & \text{if } P_{i+4} \leqq Z_i \\ 1 & \text{if } P_{i+4} > Z_i \end{cases}$$

그림 1. MCT(Modified Census Transform) 알고리즘
Fig. 1. MCT(Modified Census Transform) algorithm.

$$U_i = \begin{cases} 0 & \text{if } P_{i-3} - 8 > P_i \\ 2 & \text{if } P_{i-3} + 8 < P_i \\ 1 & otherwise \end{cases} \quad D_i = \begin{cases} 0 & \text{if } P_{i+3} - 8 > P_i \\ 2 & \text{if } P_{i+3} + 8 < P_i \\ 1 & otherwise \end{cases}$$

$$L_i = \begin{cases} 0 & \text{if } P_{i-1} - 8 > P_i \\ 2 & \text{if } P_{i-1} + 8 < P_i \\ 1 & otherwise \end{cases} \quad R_i = \begin{cases} 0 & \text{if } P_{i+1} - 8 > P_i \\ 2 & \text{if } P_{i+1} + 8 < P_i \\ 1 & otherwise \end{cases}$$

그림 2. ICT(Improved Census Transform) 알고리즘
Fig. 2. ICT(Improved Census Transform) algorithm.

$$\Gamma_i(X) = U_i\times 27 + D_i\times 9 + L_i\times 3 + R_i \quad (2)$$

그림 2와 같이 변환할 경우 그 결과값인 커널 인덱스 값은 0부터 80까지 81가지의 경우값으로 표현될 수 있으므로 필터의 크기가 상대적으로 작아 메모리에 저장하기에 적합하다. 그러나 그림 1의 MCT 변환은 주위 8개의 픽셀을 비교하였고 그림 2의 ICT 변환은 주위 4개의 픽셀을 비교하여 성능저하의 가능성이 있다. 그러나 MCT 변환은 크고 작음으로 비교를 하였고 ICT 변환은 비슷한 것과 아주 큰 것, 아주 작은 것으로 보다 자세히 비교하였고 비교되지 않는 Pi-4, Pi+4, Pi-2, Pi+2는 다음 Pi+1의 변환시 적용되므로 성능에 영향을 주지 않는다. 그림 3은 ICT의 결과로 같은 구조를 가지고 있지만 밝기가 다른 2개의 영상을 ICT 변환을 이용하여 구조정보만 추출한 결과 큰 차이가 없음을 보여주고 있다.

그림 3. ICT 변환 결과
Fig. 3. Result of ICT transform.

2. 얼굴 검출 필터 제작

본 논문에서 사용된 필터는 ICT 변환된 영상 위에서 Bernhard Fröba [1]가 제안한 학습 알고리즘을 이용하여 만들어졌으며 해당 과정은 얼굴인 영상과 얼굴이 아닌 영상에서 비교될 수 있는 모든 특징 위치마다

의 룩업 테이블을 생성하는 것이다. 특징위치는 구별자로 정의되며 각 룩업 테이블은 커널 인덱스($\Gamma_i(X)$) 값에 따른 필터값으로 구성된다.

$$H(\Gamma) = \sum_X h_X(\Gamma(X)) \tag{3}$$

$$h_X(\Upsilon) = \sum_{t=1}^{T} a_t w_t(\Upsilon) I(X = X_t) \tag{4}$$

$$a_t = \frac{1}{2} ln(\frac{1-\epsilon_t}{\epsilon_t}) \tag{5}$$

식(3), (4), (5)는 얼굴 필터를 만드는 과정 중 사용되는 수식이다. 식(3)은 특징 위치인 모든 구별자에 대한 룩업테이블을 합하여 얼굴 필터를 생성하는 과정이고 식(4)는 한 위치에 대한 룩업 테이블을 생성하는 과정이다. 식(5)에서는 a_t를 정의하며 식(5)의 ϵ_t는 각 구별자들의 각 커널 인덱스값에 따라 변화되는 변수로 필터의 실질적인 값이 된다.

3. 다중 얼굴검출 알고리즘

ICT 변환 이미지 상에서 학습되어진 필터를 적용하여 얼굴을 찾기 위해 본 논문에서 하드웨어로 구현한 얼굴 검출 알고리즘은 그림 4와 같다. 전체 흐름은 그레이스케일 입력 영상을 변환하고 변환된 이미지에 8x8 필터를 적용하여 얼굴 후보(candidate_range1)를 검출한 뒤, 검출된 영역을 16x16 필터와 20x20 필터를 통해 최종 검증하여 얼굴 영역(face_range)을 출력해주는 것이다.

```
input : 640*480 gray image
output : face_range

while (begin_ratio * mult_ratio > end_ratio) do {
  resize(ratio, original_image, resized_image)
  ict(resized_image, ict_image)
  find_cand(8*8 filter, ict_image, pre_candidate_range1)   }
group(pre_candidate_range1, candidate_range1)

repeat all candidate_range1 {
  while (begin_ratio * mult_ratio > end_ratio) do {
    resize(ratio, candidate_range1, resized_image)
    ict(resized_image, ict_image)
    find_cand(16*16 filter, ict_image, pre_candidate_range2)
    group(pre_candidate_range2, candidate_range2)   }   }

repeat all candidate_range2 {
  while (begin_ratio * mult_ratio > end_ratio) do {
    resize(ratio, candidate_range2, resized image)
    ict(resized_image, ict_image)
    find_cand(20*20 filter, ict_image, pre_candidate_range3)   }
  group(pre_candidate_range3, face_range)   }
```

그림 4. 얼굴 검출 알고리즘
Fig. 4. Face detection algorithm.

그림 4에서 resize는 입력되는 ratio에 따라 원본 이미지를 축소한다. 축소하는 방법은 1/ratio의 크기로 픽셀 선택하는 방식을 취한다. *ict*는 흑백 이미지를 ICT 변환하는 부분으로 식(2)와 같이 변환한다.

find_cand 함수는 3가지 필터(8x8, 16x16, 20x20 필터)를 적용하여 얼굴 후보 영역를 검출하며, *group* 함수는 *find_cand* 함수에서 검출한 얼굴 후보 중 크기가 비슷하고 가까운 거리에 있는 후보들을 하나의 대표자로 만드는 동작을 한다. 이것은 서로 가깝지 않거나 크기가 다른 것은 다른 얼굴로 인식하여 다중 얼굴 검출이 가능하도록 한다.

III. 하드웨어

설계된 하드웨어의 블록도는 그림 5와 같으며 대부분 소프트웨어의 함수를 하드웨어 모듈로 설계한 것을 알 수 있다. 메모리 사용에 있어서, SDRAM에는 원본 이미지(640x480 : 300KByte)와 필터(8x8, 16x16, 20x20 필터 : 1148KByte)가 저장되어 있고, 내부 SRAM의 구성 중 Cascade_info_mem는 필터 적용시 필요한 파라미터

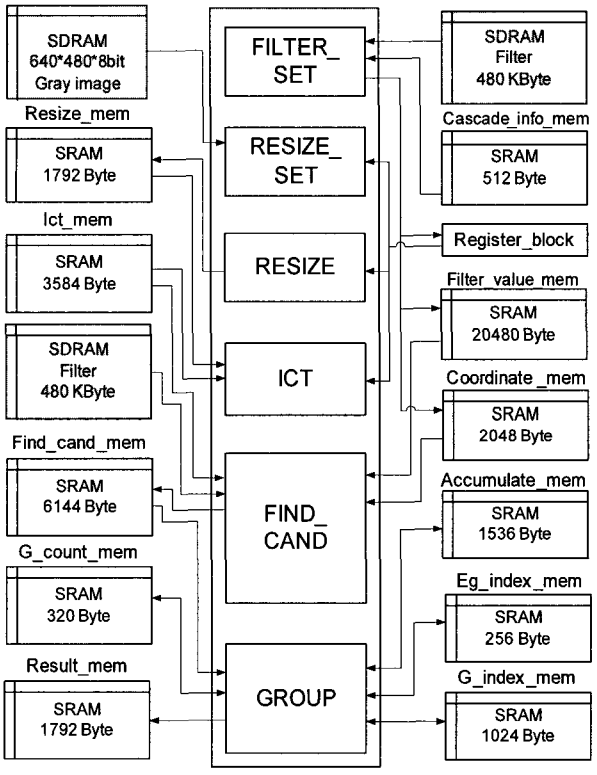


그림 5. 얼굴 검출 하드웨어 블록 다이어그램
Fig. 5. Block diagram of Face detect hardware.

값들이 저장되어 있다. Resize_mem과 Ict_mem에는 각각 축소된 영상과 ICT 변환 이미지가 저장되며, Find_cand_mem과 Result_mem에는 FIND_CAND 모듈에서 필터를 적용한 후의 얼굴 후보영역들과 이 영역들을 GROUP 모듈에서 대표 후보영역으로 그룹화한 결과가 저장된다.

각각의 모듈은 소프트웨어 과정에서 수행되어지는 함수들과 동일한 동작을 한다. 그러나 FILTER_SET은 소프트웨어 수행 과정 중 포함되지 않은 모듈로 SDRAM에 저장된 필터(8x8, 16x16, 20x20 필터)중 FIND_CAND 모듈에서 빠른 속도록 필터를 적용할 수 있도록 일부의 필터값를 SRAM에 복사하는 역할을 한다. 그리고 RESIZE_SET 모듈 역시 소프트웨어 과정 중 포함되지 않은 모듈이며, RESIZE 모듈에서 원본 이미지를 축소하기 위한 비율(RATIO, inverse RATIO)을 계산하는 모듈이다.

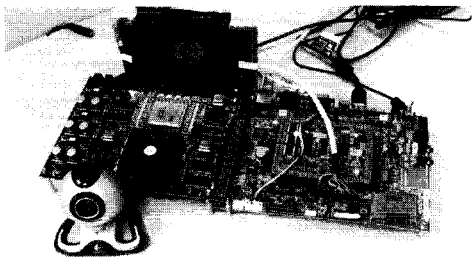


그림 6. Bluesky ARM 보드 & Xilinx Virtex4 FPGA 보드
Fig. 6. Bluesky ARM board & Xilinx Virtex4 FPGA board.

전체 하드웨어는 ASIC으로 동작속도 100Mhz를 목표로 하여 FPGA에서 75Mhz의 클럭으로 구동할 수 있도록 크리티컬 패스를 구성한다. 그리고 설계 하드웨어의 FPGA 검증을 위해 ARM920T를 내장한 Bluesky ARM 보드와 Xilinx Virtex4 FPGA와 삼성 32MB SDR SDRAM이 탑재된 보드를 일반 메모리 인터페이스 방식으로 연결하여 그림 6과 같이 구성한다.

IV. 메모리 최적화

소프트웨어에서 각 함수의 입/출력이 원본 이미지와 축소된 이미지, ICT 변환 이미지 등이기 때문에 하드웨어 설계 시 많은 양의 메모리를 필요로 한다. 따라서 하드웨어 설계 시 효율적인 메모리의 엑세스와 메모리 크기를 고려하여 설계한다. 즉 외부 메모리는 외부 메모리에서 읽은 데이터는 내부에 적당량의 버퍼를 두어 다시 읽지 않도록 설계하며 엑세스 양이 많은 데이터는 미리 내부 메모리에 복사하도록 설계한다. 그리고 내부 메모리는 연속적인 내부 메모리 엑세스가 필요할 경우 메모리의 입력값 계산을 단계로 나누어 레지스터에 저장하여 매 클럭에 내부 메모리를 엑세스 할 수 있도록 설계한다.

1. 외부 메모리의 효율적인 엑세스

가. Resize_mem

Resize_mem은 그림 5의 블록도에서 픽셀 선택 방식으로 축소된 이미지를 ICT로 변환하기 위한 중간 버퍼이다. 소프트웨어 함수 처리와 같이 Resize_mem에 축소된 영상 한 프레임을 저장하려면 축소 이미지 중 최대 크기인 320x240 크기의 영상을 저장할 큰 메모리가 필요하게 된다. 반면 ICT 변환을 위해 필요한 5개의 픽셀을 저장하기 위한 레지스터 공간만 마련한다면 외부 메모리의 엑세스 횟수가 많아진다. 따라서 하드웨어는 효율적인 메모리 엑세스 구조로 설계해야 한다.

ICT 변환을 하기 위해서는 그림 7-(a)와 5개의 원본 이미지의 픽셀값을 필요로 한다. ICT 하드웨어 모듈이 그림 7-(b)의 (1)픽셀을 ICT 변환하여 Ict_mem에 저장한 뒤, (2) 픽셀을 변환하여 Ict_mem에 저장할 때 a, b, c 픽셀값을 새롭게 필요로 하기 때문에 외부 메모리에서 a, b, c 픽셀값을 읽어온다. 이후 그림 7-(b)의 화살표 방향으로 ICT 변환 중 (3)픽셀을 변환하기 위해서 b, c, d 픽셀값이 필요하다. 그런데 이중 b, c가 (2) 픽셀을 변환할 때 사용했던 픽셀과 중복된다. 이런 중복된 데이터를 외부 메모리

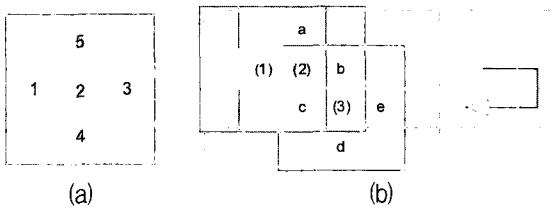


그림 7. Resize_mem 메모리의 엑세스 과정
Fig. 7. Process of Resize_mem memory access.

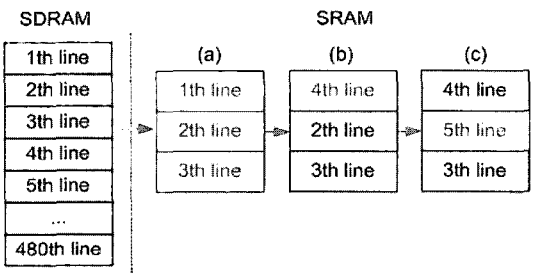


그림 8. Resize_mem 메모리 구성
Fig. 8. Resize_mem memory architecture.

에서 읽는 동작을 피하기 위해 중간 버퍼에 b, c가 속한 이미지의 두 줄의 데이터가 Resize_mem에 저장될 필요성이 있다. 즉 ICT 변환을 위해서는 원본 이미지 3줄의 데이터를 저장할 필요가 있는데 첫줄을 제외한 두, 세번째 줄은 중복사용이 된다. 따라서 중복 사용 데이터를 포함하여 ICT 변환이 가능한 최소 내부 메모리 사용을 위해 그림 8과 같이 메모리를 구성한다.

그림 8은 Resize_mem에 저장되는 데이터의 순서이다. (a) 경우와 같이 이미지의 1, 2, 3번째 줄을 저장하고 ICT 변환을 한 후 (b)의 경우가 될 때 1번째 줄은 다시 사용되지 않는다. 따라서 1번 째 줄이 저장되었던 공간에 4번째 줄을 덮어쓰고 ICT 변환을 한다. (c)의 경우도 2번째 줄이 다시 ICT 변환시 요구 되지 않기 때문에 5번째 줄을 덮어 써 ICT 변환을 한다. 이와 같이 Resize_mem의 엑세스 하드웨어를 구성하면 외부 메모리 중복 읽기를 피할 수 있으며 메모리 사용을 320x240 크기의 메모리가 아닌 320x3 크기의 메모리를 소요하여 메모리 사용을 최소화 할 수 있다. 이와 같은 구조를 Ict_mem에 엑세스하는 모듈에 확장하면 전체적으로 148 KByte의 내부 메모리를 절약할 수 있다.

나. Filter_value_mem, Coordinate_mem과 Register block

FIND_CAND 모듈은 필터를 적용하여 얼굴 후보를 검출하는 모듈이다. 필터 적용 방법은 그림 9과 같이 0부터 80까지의 값을 가지고 있는 1, 2 등의 구별자의 커널 인덱스값을 Ict_mem으로부터 읽고 각 구별자마다의 룩업 테이블과 비교하여 필터값을 얻는다. 이후 모든 구별자의 필터값을 누적하고 6단계의 케스케이드의 스레스홀드값들과 비교하여 얼굴 영역임을 판단한다.

이와 같은 방식으로 필터를 체크하기 위하여 필요한 값은 구별자의 좌표와 인덱스에 따른 룩업 테이블이다. 그러나 필터가 8x8 필터와 16x16필터, 20x20필터가 모두 사용되기 때문에 모든 필터를 SRAM에 저장할 수 없다. 따라서 모든 필터 데이터(필터값, 좌표)를 외부

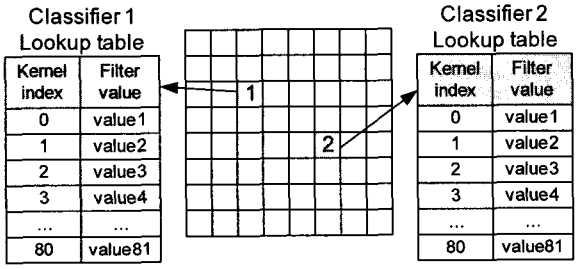


그림 9. 필터 적용 과정
Fig. 9. Filtering process.

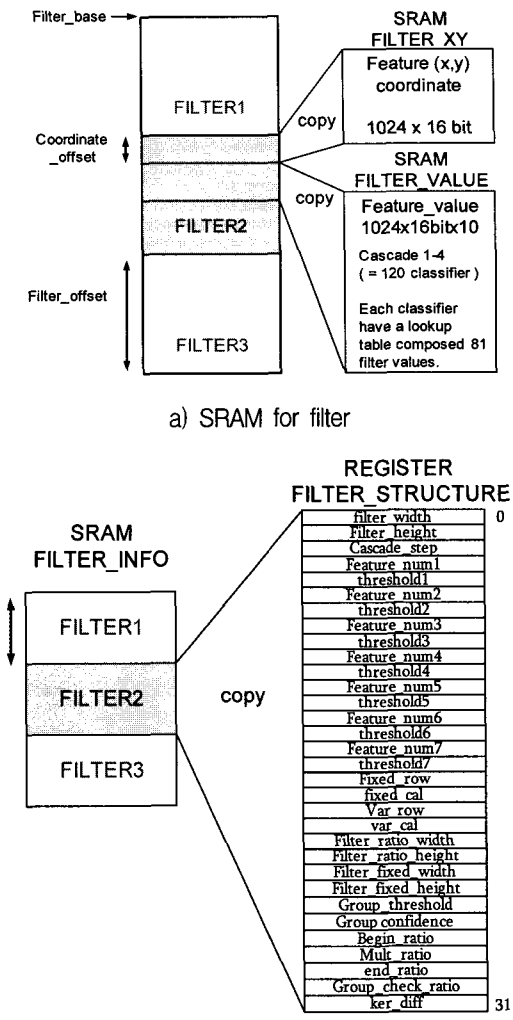


그림 10. 필터를 저장하기 위한 메모리와 레지스터
Fig. 10. Memory and Register for Filter.

메모리 SDRAM에 저장한다. 그러나 외부 메모리에서 필터 데이터를 읽어오는 것은 많은 수의 클럭을 소요하기 때문에 적용할 필터만 내부 메모리에 복사하여 클럭 소요를 최소화한다. 또한 각 필터의 정보와 각 케스케이드에 필요한 모든 변수를 Cascade_info_mem에 저장하여 적용하고자 하는 필터의 종류에 따라 필요한 데이터를 Register_block으로 복사하는 방식을 취한다.

Cascade_info_mem은 내부 메모리이지만 Cascade_info_mem의 데이터를 Register_block으로 복사하는 이유는 필터 적용 시, 데이터가 동시에 여러 값들이 사용되어야 하기 때문이다. 그림 10과 같이 각 필터 적용에 맞는 데이터를 복사하기위해 FILTER_SET 모듈이 필요하다. 사용하는 필터의 종류(8x8, 16x16, 20x20 필터)에 따른 데이터 복사는 모든 필터 체크 좌표와 필터 테이블을 저장하고 있는 SDRAM으로부터 Coordinate_mem에 필터 체크

좌표를 복사하고 Filter_value_mem에 필터 테이블을 복사한다. 또한 기타 필터 적용에 필요한 케스케이드 정보, 변수 등을 저장하고 있는 Cascade_info_mem으로부터 각 필터 적용시 필요한 데이터를 Register block으로 복사하도록 FILTER_SET 모듈을 설계한다.

그러나 필터를 적용할 때 필요한 각각의 필터 테이블의 크기도 Filter_value_mem에 저장하기에는 큰 용량을 가지고 있기 때문에 필터 적용시 다수의 메모리 엑세스를 필요로 하는 케스케이드 1, 2, 3, 4단계(120개 필터 체크 좌표)의 필터 테이블 만을 저장하고 이를 통과한 이미지는 SDRAM에 저장되어 있는 나머지 필터 체크 좌표까지 점검하여 최종 결과를 얻도록 설계하였다. 그림 10은 SDRAM에서 Filter_value_mem과 Coordinate_mem으로 복사하는 것과 Cascade_info_mem에서 Register block으로 복사하는 자세한 내용이다.

2. 내부 메모리의 효율적인 엑세스

얼굴 검출 하드웨어를 구성하는 모듈 중의 FIND_CAND 모듈이 가장 많은 클럭을 소요한다. 얼굴 검출 알고리즘(그림 4)에 따른 과정 중 find_cand 함수의 과정은 필터 체크 좌표를 읽어오고 이로부터 ICT로 변환된 이미지에서 커널 인덱스 값을 읽어와 그 값을 다시 필터 테이블의 주소로 사용하여 최종 필터값을 누적한다.

이와 같은 과정으로 하드웨어를 설계하면 여러 메모리의 출력을 임의의 로직을 거쳐 다른 메모리의 입력으로 들어가는 구성되며. 메모리와 메모리 사이의 로직에서도 연상을 위한 레지스터가 사용되기 때문에 그림 11과 같이 크리티컬 패스가 구성된다.

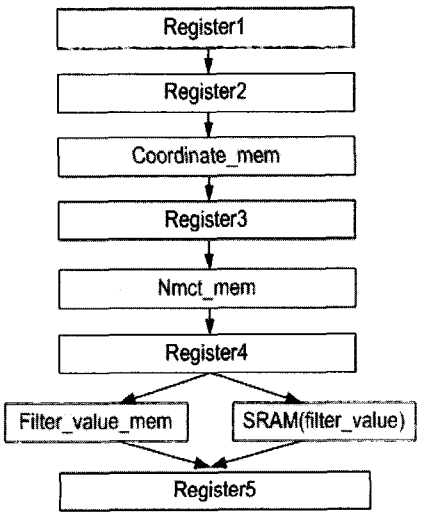


그림 11. FIND_CAND 모듈의 데이터 패스
Fig. 11. Data path of FIND_CAND module.

그러나 이와 같은 데이터 패스는 하나의 필터 체크 좌표의 필터값을 누적하기 위해서는 많은 수의 클럭이 소요된다. 즉 1024개의 필터값을 누적하려면 1024 x 8 클럭이 소요된다. 따라서 이를 파이프라인 구조를 적용하여 크리티컬 패스가 아닌 패스도 레지스터를 위치시켜 매 클럭마다 누적값이 누적될 수 있도록 한다. 즉 매 클럭마다 매 레지스터를 다음 필터를 위한 준비단계로 사용될 수 있도록 구성하여, 그림 11의 Register2의 경우 매 클럭마다 새로운 Coordinate_mem 메모리의 주소를 출력하고 Register4는 매 클럭 당 SDRAM, 또는 내부 메모리 Filter_value_mem 메모리의 주소를 출력할 수 있도록 한다.

이와 같은 파이프라인 구조를 적용하여 내부 메모리에서 필터값을 읽어 올 때, 매 클럭에 그림 11의 Register5에서 필터값을 누적하여 1024개의 필터를 8 + 1023의 클럭 소모를 통해 누적할 수 있다.

Ⅳ. 비교 분석 및 결과

본 논문에서 설계한 하드웨어는 삼성 0.18um공정을 통해 칩으로 생산하는 중이며, 그림 12는 본 하드웨어의 칩 레이아웃 사진이다. 공정이 Pad 바운더리 형식의 공정이기 때문에 실제 80만 게이트의 코어 크기에 비해 여유 공간이 있다. 따라서 파워를 충분히 공급하기 위해 파워링의 두께를 두껍게 하여 여유 공간을 활용하였고. 메모리의 위치는 각 모듈에서 사용하는 메모리를 가급적 가까운 곳에 위치시켰다.

제안된 하드웨어의 성능 분석을 위해, 기존에 설계된 하드웨어 중 정확한 성능이 기술된 이수현의 얼굴 검출을 위한 SoC 하드웨어 구현 및 검증[5]의 성능과 비교하였다.

본 논문의 하드웨어는 다른 얼굴검출 하드웨어에 비해 BioID 얼굴 DB를 통한 얼굴 검출률 측정 시 그 성능이 향상되었고 효율적인 메모리 엑세스 설계로 메모리 사용이 적다. 동작 클럭은 최대 100Mhz로 동작 하며 최대 얼굴 검출 개수는 20개로 다중 얼굴 검출이 가능하다. 입력영상 크기는 다른 하드웨어의 160x120 크기에 비해 최대 640x480 크기까지 동작 되고 테이블 1에 명시된 크기 이외에도 임의의 여러 종류의 영상 크기를 입력할 수 있다. 또한 입력영상으로 320x240 크기의 입력영상을 사용하고 100Mhz로 동작시킬 경우 1초에 25 프레임이 처리되어 실시간 처리가 요구되는 어플리케이션에 적용 가능하다.

표 1. 성능 분석
Table 1. Analysis of performance.

비교항목	참고문헌[5]	본 논문	
얼굴검출 알고리즘	Haar-Like	ICT	
입력 영상 크기	160x120	640x480	320x240
하드웨어 처리 속도(100Mh)	2.04 f/s	5 f/s	25 f/s
얼굴 검출 갯수	1개	20개	
Max Clock	100Mh	100Mh	
SRAM 크기	88KB	40KB	
SDRAM 크기	1 MB	1.6MB	
로직크기	약 300,000 gate	약 800,000 gate	
얼굴 검출률 (BioID 적용)	97%	98%	

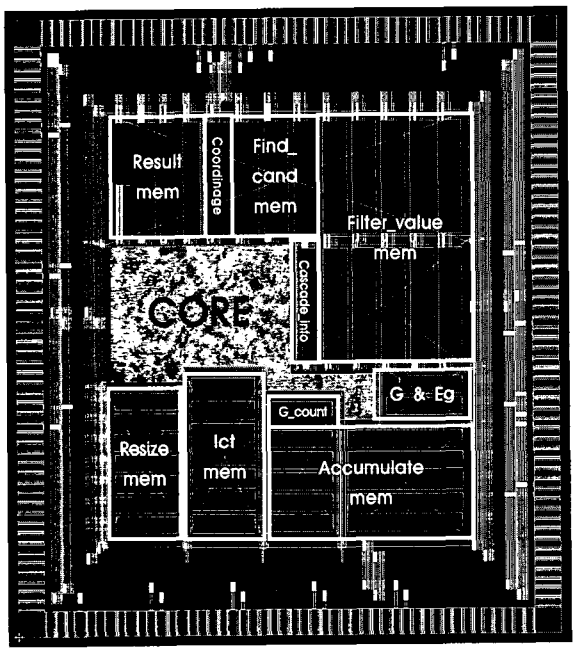


그림 12. Chip 레이아웃
Fig. 12. Chip Layout.

V. 결 론

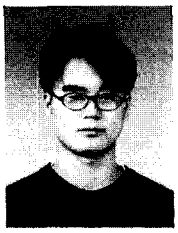
본 논문은 얼굴 인식의 기본이 되는 얼굴 검출을 여러 임베디드 환경의 어플리케이션에 적용할 수 있도록 하는 것에 목적이 있다. 따라서 ICT 변환 알고리즘을 바탕으로 한 얼굴 검출 알고리즘을 효율적인 외부, 내부 메모리의 엑세스를 고려한 하드웨어로 설계하여 그 목적에 부합하였다.

설계된 하드웨어는 40KByte의 내부 메모리와 약 80만 게이트의 로직으로 구성되었으며 실시간 처리가 가능하고, BioID를 통해 성능을 측정한 결과 98%의 검출률을 나타내었다. 따라서 본 논문에서 제안한 하드웨어는 차세대 생체인식 기술인 얼굴 인식 기술에 접목되어 인공지능 로봇, 보안시스템, 스마트 토이 등 실제 어플리케이션에 적용가능 할 것으로 예상된다.

참 고 문 헌

[1] Bernhard Fröba, Andreas Ernst, "Face detection with the modified census transform," in Proc. of IEEE Conf. on AutoSmatic Face and Gesture Recognition, pp. 91-96, May 2004.
[2] 이호근, 정성태, "실시간 얼굴 검출 시스템 설계 및 구현," 한국 멀티미디어 학회, 8권, 제8호, 2005년 8월.
[3] 신승주, 최석림, "HCr과 적응적 임계회에 의한 고속 얼굴 검출," 전자공학회논문지, 제41권, 제6호, 2004년 11월.
[4] 박상근, 박영태, "신경회로망 및 기하학적 특징을 이용한 얼굴영역 검출," 한국정보과학회 봄 학술발표논문집 Vol. 30, No. 1, pp. 298-300, 2003년 4월.
[5] 이수현, 정용진 "얼굴 검출을 위한 SoC 하드웨어 구현 및 검증", 전자공학회논문지, 제44권, 제4호, 2007년 4월

저 자 소 개



김 윤 구(학생회원)
2006년 광운대학교 전자공학부 학사 졸업.
2006년 3월～현재 광운대학교 전자통신공학과 석사과정
<주관심분야 : SoC 설계, 영상처리 및 인식, 임베디드 시스템 설계>

정 용 진(정회원)
1983년 서울대학교 제어계측공학과 학사 졸업.
1983년 3월～1989년 8월 한국전자통신연구원.
1995년 미국 UMASS 전자전산공학과 박사 졸업.
1995년 4월～1999년 2월 삼성전자 반도체 수석연구원.
1999년 3월 광운대학교 전자통신공학과 부교수
<주관심분야 : 무선통신, 정보보호, SoC 설계, 영상처리 및 인식, 임베디드 시스템>