임베디드 생체인식기술 구현: 지문 보안토큰 시스템

Implementation of Embedded Biometrics Technologies: a Security Token System for Fingerprints

김영진(Y.J. Kim)	생체인식기술연구팀 연구원
반성범(S.B. Pan)	생체인식기술연구팀 선임연구원
문대성(D.S. Moon)	생체인식기술연구팀 연구원
길연희(Y.H. Gil)	생체인식기술연구팀 연구원
정용화(Y.W. Chung)	생체인식기술연구팀 책임연구원, 팀장
정교일(K.I. Chung)	정보보호기반연구부 책임연구원, 부장

지문 정보 등의 생체 정보를 이용하는 생체 기술은 컴퓨터 시스템의 로그인, 출입 ID, 전자상거래 보안 등의 여러 서비스에서 사용자의 안전한 인증을 위해 널리 사용되고 있다. 근래에 이르러, 생체 기술은 비밀 번호와 같은 기존의 개인 인증 방법에 비해 안전하면서도 자동화를 가져올 수 있다는 장점으로 인해 보안 토큰, 스마트 카드와 같은 소형의 임베디드 시스템에 탑재되고 이용되는 추세이다. 본 논문에서는 보안 토큰을 이용한 생체 인식 기술의 시장 동향을 살펴보고 임베디드 시스템의 형태인 보안 토큰 시스템을 개발하고 시험한 결과를 기술하였다. 보안 토큰과 호스트와의 통신은 USB를 이용하여 시험 및 검증하였으며 보안 토큰 상에서의 지문 정합 프로그램의 성능 측정 및 개선에 대해 기술하였다. 나아가, 보안 토큰에서 매치 온 카드(match-on-card)로의 전이를 위해 필요한 내용을 언급하였다.

I. 서 론

컴퓨터 시스템 로그온이나 출입 ID에 더하여 근래에는 전자상거래, 전자화폐 등의 폭넓은 사용에 따라 개인 인증 처리가 많아지고 있어, 개인 정보의 해킹에 대한 위협도 커지고 있다. 따라서, 개인 정보의 안전한 관리 및 안전한 사용자 인증이 요구되고 있는 실정이다. 현재 주로 사용되고 있는 개인 인증 방법인 패스워드나 개인 번호(PIN)의 사용은 해킹 또는 관리 부주의에 따른 노출이 쉬워서 보다 안전한 사용자 인증 방법이 필요하다. 보안 토큰 시스템은 지문, 얼굴, 음성 등의 생체 정보를 개인 식별을 위한 인증 자료로서 이용하는데, 본 논문에서는 일종의 임베디드 시스템인 match-on-token을 지칭하도록 한다. Match-on-token은 기준이 되는 사용자 생체 정보를 미리 토큰 내에 저장하고, 인증이 필요할 때 외부에서 생체 정보를 받아 들여 토큰 내에서 정합(matching)을 수행한다. 이로써, 호스트를 포함한 토큰 외부에서 일어날 수 있는 사용자 정보에 대한 해킹을 방지하게 된다. 또한, match-on-token은 스마트 카드를 이용한 match-on-card로 연계되어 발전하고 있다. 또한, 보다 안전성이 강화된, 카드에 지문 입력 센서를 부착한 sensor-on-card도 개발되고 있는 추세이다.

본 논문에서는 match-on-token을 포함하는 보안 토큰 시스템 개발 내용에 대해 기술하였다. 먼저, Ⅱ장에서 현재 시장에서 사용되고 있는 보안 토큰 및 보안 토큰을 이용한 생체 인식 기술 동향을 살펴보고, Ⅲ장에서는 보안 토큰 에뮬레이터 보드와 호스트를 중심으로 하는 보안 토큰 시스템의 구성 요소 및 각 요소별 설계 사항에 대해 언급하였다. Ⅳ장에서는 보안 토큰 시스템의 구현 및 시험 내용에 대해 기술하였다. 또한, 지문 정합 프로그램의 성능 측정에 대한 내용을 살펴보았다. 끝으로, Ⅴ장에서 성능 개선 및 match-on-card로의 전이를 위해 필요한 내용 등의 향후 연구 방향에 대해 언급하였다.

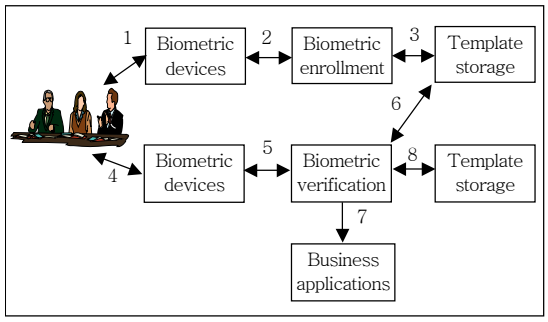
Ⅱ. 보안 토큰을 이용한 생체인식 기술 동향

생체정보를 이용한 사용자 인증 기술은 지문과 같은 생체정보가 개인별로 고유한 특징임이 증명된 이후부터 계속적으로 사용자 인증에 사용하려는 연구가 진행되어 왔다. 그리고 이러한 연구가 실생활에 적용되기 시작한 것은 지문의 경우 광학식 또는 반도체식 지문 장치가 개발되고, 지문인식에 필요한 많은 계산을 실시간으로 처리할 수 있는 고성능 컴퓨터가 일반 사용자에게 보급된 1990년대 이후부터이다. 그리고 앞으로의 사용자 인증은 CPU를 내장한 보안 토큰만을 가지고서도 생체정보를 이용한 사용자 인증이 가능하게 될 것으로 예상된다.

일반적으로 생체정보를 이용한 인증을 수행하는 과정은 (그림 1)과 같이 생체정보취득, 생체인증연산, 생체정보저장 과정으로 나눌 수 있다.

또한 (그림 2)에 나타낸 생체정보를 이용한 보안 토큰은 보안 토큰 내에 메모리만 있는 경우, 연산 프로세서도 있는 경우, 센서까지 있는 경우에 따라 store-on-token, match-on-token 및 sensor-on-token으로 나눌 수 있다.

Store-on-token 방식은 지문과 같은 생체정보를 중앙 집중식 DB에 저장하지 않고 보안 토큰 내의 메모리에 저장한 후 인증을 요청할 시에 저장된



(그림 1) 생체인증 과정

(a) 스마트 카드 (b) USB 토큰

(그림 2) 생체인증을 위한 보안 토큰 예

생체정보를 단말에 보내어 단말기에서 인증을 하는 시스템이고, match-on-token은 저장된 생체정보와 인증을 요청할 시에 취득한 생체정보를 보안 토큰에서 인증 알고리듬을 수행하여 정합한 뒤에 보안 토큰에서 인증 결과만을 단말쪽으로 보내는 방식이다. 그리고 위의 두 종류의 토큰에서 생체정보 획득은 단말기에서 이루어지는 반면, <표 1>에 나타낸 것과 같이 sensor-on-token은 생체정보 획득 자체가 보안 토큰상에서 이루어진다는 것이다. 즉, sensor-on-token은 지문 인식기와 같은 생체 정보 입력기가 토큰에 포함되어 있어서 생체 정보의 영상 및 특징점을 추출하는 과정 자체가 토큰상에서 수행되므로 보다 안전한 생체 인증 시스템 구축이

<표 1> 보안 토큰

	생체정보 취득	생체인증 연산	생체정보 저장
Store-on-Token	×	×	○
Match-on-Token	×	○	○
Sensor-on-Token	○	○	○

가능하다.

생체인식과 스마트 카드를 결합하는 대표적인 연구로는 유럽 IST(Information Society Technologies)에서 추진하고 있는 Biometric Matching and Authentication System on Card 프로젝트이다. 2001년 1월부터 2002년 6월까지 18개월간 수행된 프로젝트의 목적은 생체인식을 스마트카드에 결합하여 보다 안전하고 편안한 인증 및 인식 방법을 제공하는 것이다. 생체와 스마트카드는 여러 레벨에서 결합될 수 있으며, 기존 시스템 보다 높은 보안성과 프라이버시를 제공할 수 있다.

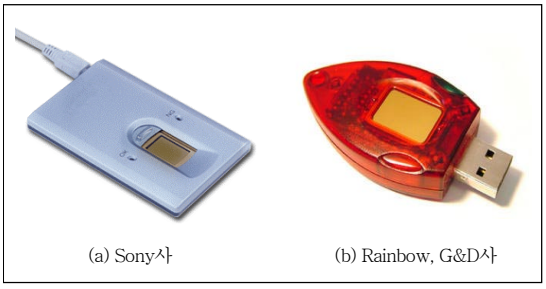
수행방법은 안전한 생체 스마트카드 시스템을 실현하기 위하여 biodata-on-card, matching-on-card, encoding-on-terminal 기술과 생체에 적합한 칩카드 프로세서 등을 개발한다. 그리고, 개발된 시스템은 e-commerce, health care, access control, data security & encryption 등의 응용분야에서 시험한다.

Store-on-token 방식의 기술 개발 사례는 세계적인 생체인식 업체인 Veridicom사에서 자사의 지문 인식 시스템을 이용한 store-on-card 방식의 스마트 카드를 개발하여 PC 및 인터넷 액세스 제어용으로 판매하고 있으며, 세계적인 스마트 카드 업체인 Bull사는 Keyware사의 화자 인증 시스템을 이용한 store-on-card 방식의 스마트 카드 개발을 1997년에 시작하였고, Motorola사도 Identix사와 공동으로 store-on-card 방식의 지문 인식 시스템과 스마트 카드와의 연계 기술을 개발하고 있다.

또한, 개인의 생체정보를 스마트 카드에 저장할 뿐만 아니라 스마트 카드 내의 프로세서를 이용하여 인식 처리까지 수행함으로써 개인의 정보가 보안 토큰 외부로 유출되지 않는 match-on-card 기술 개발이 현재 활발히 진행중이다. Gemplus사는 Biometric Identification사 및 Precise Biometric사와 공동으로 지문 인증 방식을 적용한 store-on-card 방식의 스마트 카드 솔루션을 바탕으로 카드 내에서 인식 처리를 수행하는 match-on-card 기술을 현재 개발중이다. 또한 (그림 3)과 같이 Obethur Card System사는 id3 semiconductors사와 공동으로 최근 스마트 카드와 카드 리더로 구성된 지문 인증 시제품을 발표하였다. 즉, 카드 리더에 있는 지문 입력 센서를 통하여 지문을 입력 받아 특징을 추출한 후 스마트 카드에 지문 특징 정보를 저장한다. 그리고 스마트 카드에서 매칭을 수행하여 인증 결과를 출력하도록 되어 있다.

(그림 4)의 (a)는 소니사의 지문 인증 시스템으로 지문 정보 저장과 지문 인증 연산을 시스템 내에서 수행하는 것으로 USB 방식으로 호스트와 통신하도록 되어 있다. 일반 PC 등의 호스트가 일반적으로 USB를 지원하므로 추가 비용 부담 없이 사용할 수 있는 장점이 있어 USB 보안 토큰 시스템 개발도 활발히 진행되고 있다. (그림 4)의 (b)는 Rainbow사와 G&D사가 개발중인 sensor-on-token으로 내부 CPU는 ARM7을 사용하고 있다.

(그림 3) Match-on-Card 시스템(Oberthur Card System사)



(a) Sony사 (b) Rainbow, G&D사

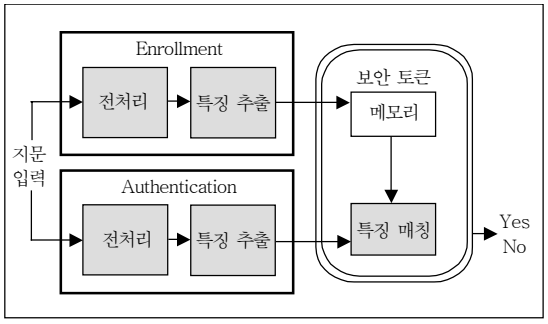
(그림 4) Sensor-on-Token

Ⅲ. 보안 토큰 시스템 설계

일반적인 보안 토큰 시스템은 서론에서 언급한 바와 같이 생체 정보를 이용하여 시스템 내에서 사용자 인증을 수행하므로 인증 과정상의 안전성을 보장하게 된다. 즉, match-on-token 상에서 생체 정보 정합을 수행한다. 본 논문에서의 보안 토큰 시스템에서 사용되는 생체 정보는 지문으로 한정하도록 한다.

지문을 이용한 보안 토큰 시스템은 (그림 5)와 같이 사용자 등록(enrollment) 과정과 사용자 인증 과정(authentication)으로 이루어진다[1]. 사용자 인증 과정은 지문 특징점의 정합을 포함한다.

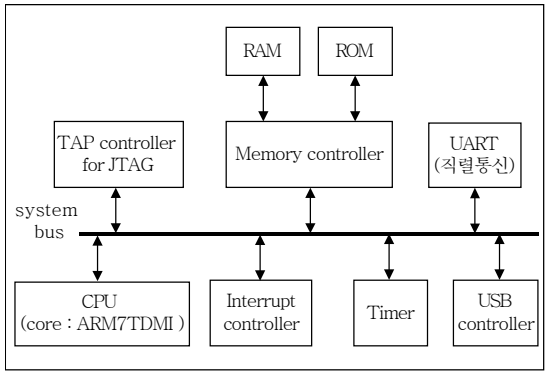
등록 및 인증 과정에서 입력된 지문 영상을 이용한 특징점 정보 추출과정까지는 호스트에서 수행한다. 등록 및 인증 지문 특징점 정보는 USB 통신 채널을 통해 보안 토큰에 저장한 후에는 외부로 전달되지 않고 개인 기기 내부에서 정합 과정까지 수행하여 인증 결과만을 외부로 출력하도록 한다. 보안 토큰의 메모리 크기 및 CPU 연산 성능에 따라 보안 토큰의 지문 정합 수행 성능이 달라지므로 이에 대한 고려가 반드시 필요하다. 본 장에서는, 보안 토큰 에뮬레이터 보드에서의 H/W 사양 및 S/W 수행 환경 설계, 호스트 프로그램의 설계, USB 통신 및 지문 정합 프로그램의 수행에 대해 기술한다.



(그림 5) 지문을 이용한 Match-on-Token 시스템

1. 보안 토큰 에뮬레이터 보드

보안 토큰은 에뮬레이터 보드의 형태로 구현하며



(그림 6) 보안 토큰 에뮬레이터 보드 구조

ARM7TDMI를 코어로 하는 상용 프로세서를 CPU로 사용한다. 메모리 뱅크를 이용하여 메모리 맵 설정 및 초기화를 하며 소스 레벨의 디버깅을 위해 JTAG을 이용한 AXD 디버거를 사용하도록 하고 상위 레벨의 간단한 디버깅 또는 모니터링을 위해 직렬 통신을 사용한다. 또한, 보드와 호스트 사이의 통신을 위해서 USB 통신을 지원하도록 한다. 이러한 내용을 포함한 보안 토큰 보드의 H/W 구조를 나타내면 (그림 6)과 같다.

보안 토큰 보드에서의 수행 S/W는 H/W와 직접 인터페이스하는 장치 구동 루틴을 포함하고 이용하는 부팅 코드와 응용 프로그램으로 나뉜다. 장치 구동 루틴은 메모리 설정 관련 레지스터에 대한 동작, 직렬 통신 장치 및 USB 장치에 대한 초기화 및 인터럽트 처리 함수 등이 포함된다. 이를 수용하는 부팅 코드 및 응용 프로그램의 수행 내용은 다음과 같이 구성된다.

① 메모리(SDRAM 등) 설정 및 초기화
② 관리자 모드 설정
③ 인터럽트 제어 레지스터 초기화 및 설정
④ 전역 코드 영역 및 데이터 영역 초기화
⑤ 각 모드별 스택포인터 설정
⑥ 예외 벡터 테이블 설정
⑦ 사용자 모드 설정 및 스택포인터 설정
⑧ 응용 프로그램(지문정합 프로그램)의 메인함수 호출

2. 호스트 프로그램

보안 토큰 보드는 호스트에서 동작하는 클라이언트 프로그램에서 넘겨주는 데이터에 대해 지문 정합을 수행한다. 이 데이터는 기준이 되는 template 지문 특징점 정보와 인증을 위해 입력을 받는 input 지문 특징점 정보이다. 이 지문 특징점 정보들은 호스트에 연결된 지문 스캐너에 연결되어 영상 이미지로부터 특징점 추출을 담당하는 프로그램으로부터 얻어지게 된다.

호스트측에서의 USB 통신을 위한 프로그램은 USB 허브를 제어하고 USB 1.1 spec.에 맞도록 통신 내용을 제어하는 드라이버와 상위 클라이언트 프로그램으로 나뉜다. USB 드라이버는 Microsoft에 의해 배포된 Windows Driver Development Kit (DDK)를 이용, 수정하여 재작성 할 수 있다[2].

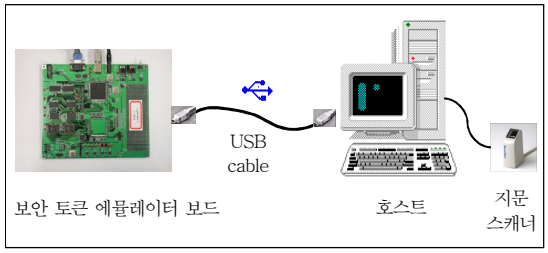
3. 지문 정합 프로그램

지문 정합 프로그램은 보안 토큰의 운영 프로그램에 기반하여 USB 통신을 통해 전달되는 template 지문 특징점과 input 지문 특징점을 입력으로 하여 정합을 수행하도록 한다. 지문 정합 프로그램은 특징점 정보의 정렬(alignment)와 정합의 모듈로 구성되며 수행된 결과는 지문의 일치 또는 불일치를 판별할 수 있는 숫자로 호스트에 반환하도록 한다. 특히, 수행 메모리의 제한을 고려해서 특징점 저장 자료 구조는 전역 변수로 하며, 불필요한 반복 연산을 줄이기 위해 루프 내에 적절한 탈출 조건을 제시하도록 한다.

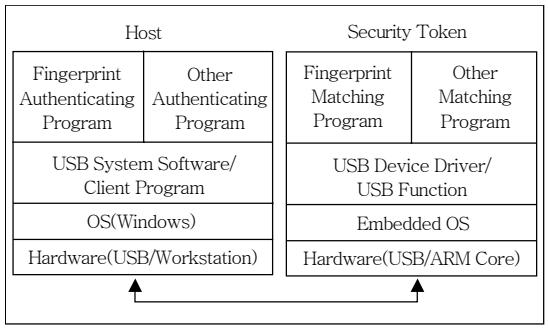
IV. 보안 토큰 시스템 구현 및 시험

1. 보안 토큰 시스템 구현

현재 개발중인 보안 토큰 시스템은, ARM 코어를 내장한 마이크로프로세서 에뮬레이터 보드상에서 ADS(ARM Developer Suite) 1.1로 부팅 코드 및 시스템 소프트웨어를 구현하고 있으며 호스트상에서는 Windows 98 운영체제를 기반으로 MS Visual C++ 6.0으로 클라이언트 환경을 구축하였다. (그림 7)은 구축된 USB 이용 보안 토큰 시스템의 구성을 보이고 있다. (그림 8)은 구현된 보안 토큰 시스템의 양대 구성 요소인 보안 토큰 보드와 호스트의 프로토콜 스택을 나타내고 있다.



(그림 7) 보안 토큰 시스템 구성



(그림 8) 보안 토큰 시스템 프로토콜 스택

에뮬레이터 보드는 ARM7TDMI를 코어로 하는 마이크로프로세서인 S5N8946을 사용하며[4], 부팅 롬으로 flash ROM 256kbytes, 프로그램 수행 공간으로 SDRAM 16Mbytes를 가진다. 특히, 메모리에 대해 보드의 하드웨어 사양과 보안 토큰의 구현 스펙을 정리하면 <표 2>와 같다. 구현 스펙의 메모리는 하드웨어적인 메모리를 부팅 코드에서 제한하고 응용 프로그램을 최적화함으로써 만족시키도록 하였다. 이것은 실제 상용 보안 토큰과 비슷한 메모리 경량

<표 2> 보안 토큰 구현 스펙

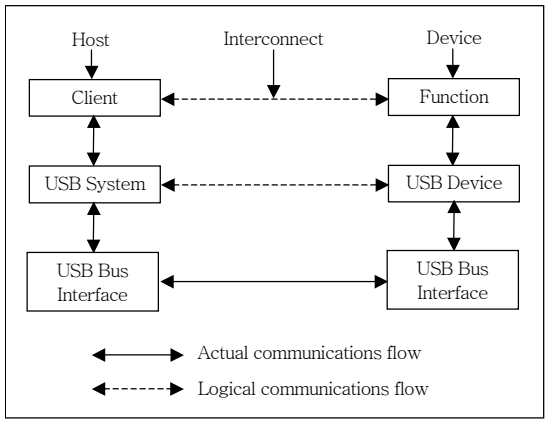
항목 / 메모리	하드웨어	구현 스펙
ROM	256k	128k
RAM	16M	64k

화를 고려한 구현 스펙이다.

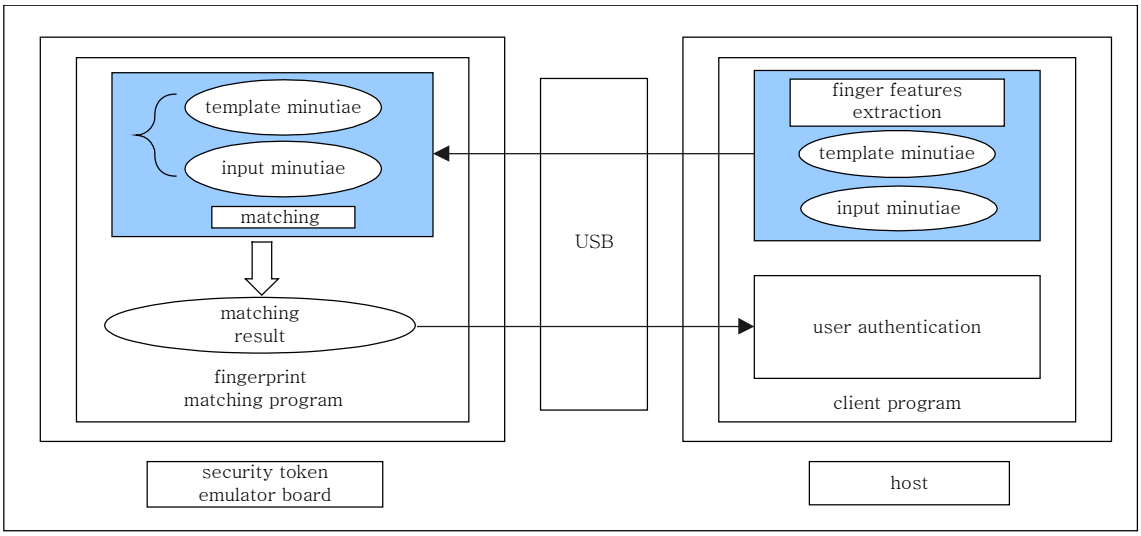
또한, 보드는 디버깅 및 모니터링을 위한 직렬 통신 포트, 지문 추출 정보의 송수신과 이를 이용해 사용자 인증을 하기 위한 USB 컨트롤러 및 허브를 가지고 있다. 부팅 코드는 SDRAM의 인식 설정 및 초기화를 시작으로 관리자 모드 및 인터럽트 모드에서의 스택포인트 설정, 인터럽트 제어 레지스터 초기화와 예외 벡터 테이블의 RAM 설치 등을 포함한다.

USB 소프트웨어는 1.1 스펙에 따라 호스트의 드라이버와 bus enumeration을 수행한 뒤에 인터럽트에 의해 송수신을 하도록 구현되었다. USB 통신을 위한 호스트 프로그램은 보드의 USB 디바이스 드라이버에서 지원하고 있는 bulk 및 interrupt transfer 타입 통신을 모두 지원하도록 구현되어 있다. (그림 9)는 호스트와 보안 토큰에서의 USB 구성 계층 간의 통신 모델을 나타낸 것이며, 최상위에 존재하는 계층, 즉 클라이언트 프로그램과 보안 토큰 상의 USB 함수가 구현되어 사용되고 있다 [3]. (그림 10)은 지문 정합 프로그램과의 USB 통신을 고려한 지문 정보를 이용하는 보안 토큰 시스템의 동작을 개략적으로 보인 것이다. 이 동작도를 기반으로 실제 보안 토큰 시스템의 수행 절차를 간략히 나타낸 내용이 (그림 11), (그림 12)에 나타나 있다.

한편, 보드에서 사용되는 지문 정합 프로그램은 한국전자통신연구원의 생체인식기술연구팀에서 개발한 피라미드 기법을 적용하여 제한된 수행 메모리를 사용하여 지문 정합을 수행하도록 하고 있다 [1],[5]. 피라미드 알고리듬은 기존 방식에 비해 지문 특징점 정합 시에 특징점의 구성 요소인 x, y 및 각도 변위에 대해 각 특징점에 대한 비교 단위를 단계적으로 조정하여 비교함으로써 사용되는 수행 메모리의 크기를 줄이고 있다.



(그림 9) USB 통신 모델

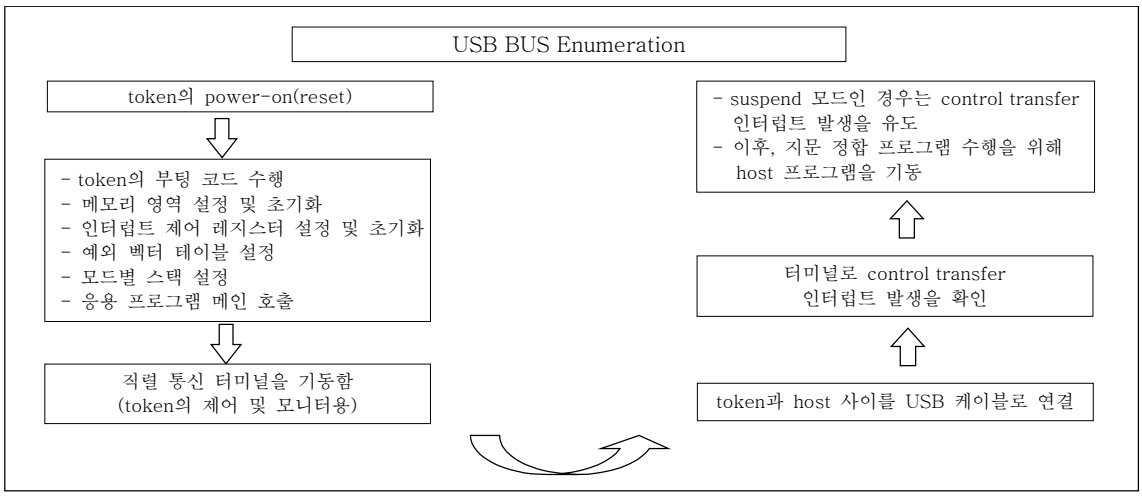


(그림 10) 보안 토큰 시스템 동작도

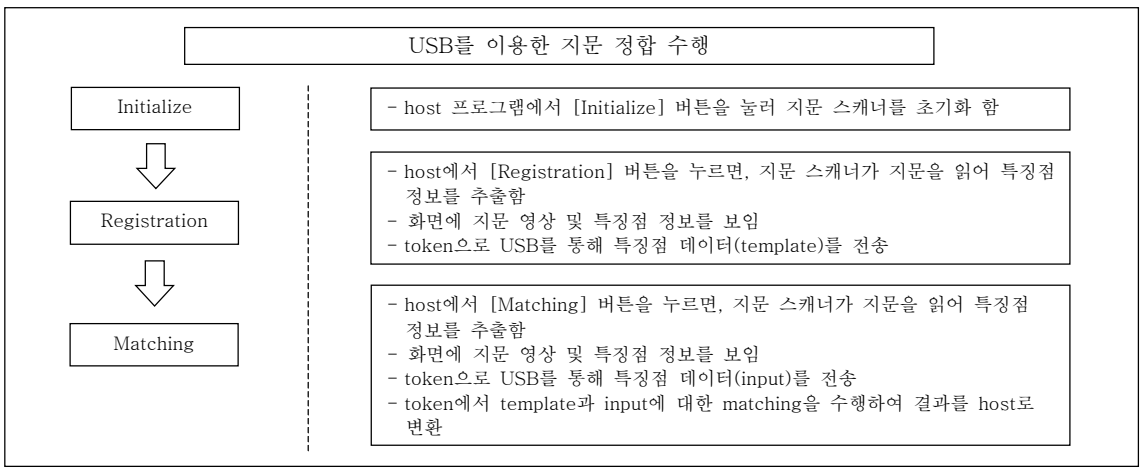
2. 보안 토큰 시스템 시험

보안 토큰 시스템의 시험 절차는 크게 호스트에서의 USB 장치 설치 및 인식 과정과 USB 설치 이후의 지문 정합 프로그램 수행으로 나뉜다. (그림 11), (그림 12)는 이러한 과정을 보여주고 있다. (그림 11)은 보안 토큰 보드에 전원이 가해지고 난 뒤에 보드의 USB 장치 드라이버를 구동하는 부팅 코드와 호스트 프로그램 간에 USB를 사용하기 전에 설치하고 인식하기 위해 통신하는 단계이다. 호스트에 보드의 USB 정치에 대한 정보가 등록되고 이에 대한 주소가 할당된 뒤에 보드에게 통보되면 USB를 이용한 통신이 가능하게 된다. USB를 이용한 보안 토큰상의 지문 정합 시험은 (그림 12)의 수행 절차에 따라 (그림 7)의 보드와 호스트 PC를 연결하여 실시하였다.

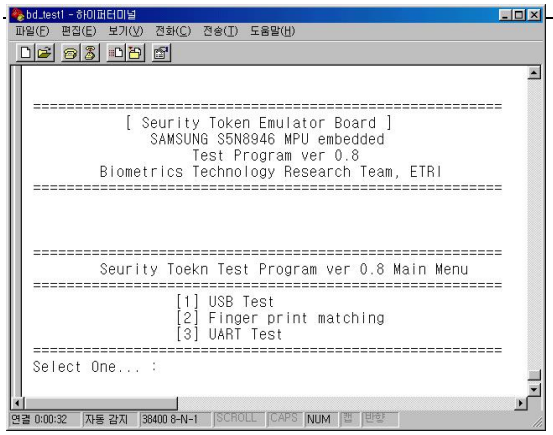
USB 설치 이후, 지문 정합 수행시 Initialize 단계에서는 지문 스캐너를 초기화하며 지문 정합에 필요한 변수들을 초기화한다. Registration 단계에서는 지문 스캐너에 입력된 지문 영상으로부터 특징점 추출 프로그램이 지문 특징점 정보들을 추출한다. 이 정보들을 가지고 있다가 정합 수행에 이용하게 된다.



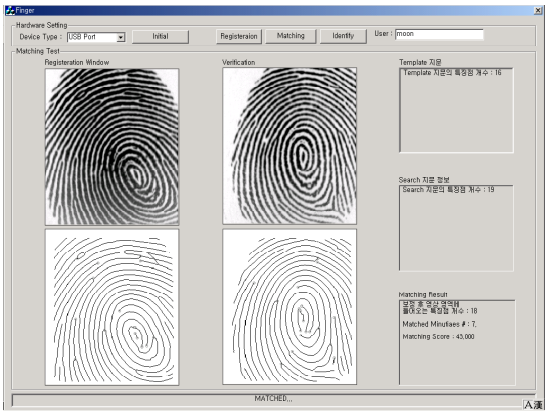
(그림 11) 보안 토큰 시스템에서의 USB 설치



(그림 12) USB 이용 지문 정합 수행



(그림 13) 보안 토큰과의 직렬 통신을 위한 터미널 화면



(그림 14) 호스트 프로그램 화면

이때, 기준이 되는 지문 정보는 보통 15~20개의 특징점으로 구성되는데 하나의 특징점은 x 변위, y 변위, 각도 변위, 방향 및 특징점 종류로 구성된다. 또, 입력 특징점 정보는 같은 사용자의 특징점이나 기준 정보에 대해 특징점 구성요소들의 값들이 상이한 15~20개의 특징점으로 구성되어 있다. (그림 13)은 직렬 통신을 통해 보안 토큰에서의 프로그램 수행을 제어할 수 있는 터미널 화면을 보이고 있으며, (그림 14)는 매칭 단계에서, 호스트 프로그램이 앞에서 기술한 지문 특징점 정보 2쌍의 정보를 보드로 USB를 통해 송신하고 정합 결과를 다시 수신하여 그 결과를 보이는 모습을 나타내고 있다.

3. 수행 시간 측정

보안 토큰 보드상에서의 지문 정합 프로그램의 수행 시간은 timer interrupt를 이용하여 측정하였다. 측정하는 내용은 interrupt 횟수와 현재의 timer counter 값이며 이를 이용하여 다음과 같이 수행 시간을 계산한다.

$$elapsed\ time = 1\text{주기의 시간} \times \left[interrupt\ \text{수} + \left(1 - \frac{counter\ \text{값}}{timer\ \text{초기값}} \right) \right]$$

여기서, timer 초기값은 interrupt 구간 길이를 지정해주는 값이다. 2절의 시험 데이터에 대해, 1주기 시간이 1초인 경우에 수행 시간 측정 값은 약 1.9초로 나타났다.

4. 수행 메모리 측정

수행 메모리를 측정하기 위해서 부팅 코드와 라이브러리 및 지문 정합 프로그램 코드의 수행 이미지가 RAM에서 이용하는 전역 변수를 확인하였다. 그 다음, 전체 RAM 메모리 한계를 제한한 뒤에 사용자 heap 메모리를 초기화하여 크기를 조절하면서 시험하였다. 부팅 코드에서는 지문 정합 프로그램이 사용자 heap을 사용하기 전에 0으로 초기화 하여 heap의 최대 사용 크기를 알 수 있도록 하였다.

2절의 시험 데이터에 대해 사용자 heap을 중심으로 측정된 수행 메모리 및 코드 크기는 다음과 같다.

- RAM 영역: 6.6kbytes(전역 변수 = 16bytes, heap = 6k, 사용자 stack = 600bytes)
- ROM 영역: 43kbytes (정합 코드 크기 = 43k)

5. 측정 성능 비교

측정한 지문 정합 수행 시간 및 수행 메모리에 대해 [5]의 시스템에서의 수행 결과와 비교한 내용은 <표 3>에서 보는 바와 같다. [5]의 시스템에서는 약 60MIPS 즉, 65MHz의 CPU[6]를 사용하고 있는 반면, 보안 토큰 보드에서는 12MHz의 클록을 사용하고 있다. CPU 성능의 차이에 기인한 수행 시간은

<표 3> 지문 정합 수행 시간 및 메모리 비교

항목 / 시스템	수행 시간(sec)	수행 메모리(bytes)
보안 토큰	1.9	6.6k
ISAVE[5]	0.29	10k

보안 토큰 보드에서 더 많이 요구되는 반면에, 수행 메모리는 보안 토큰 보드가 더 작게 사용하는 것으로 나타났다. 같은 정합 프로그램에 대해 수행 메모리가 달라진 것은, 메모리 관리에 대한 시스템 프로그램의 최적성이 보안 토큰 상에서 더 잘 이루어져 있다고 볼 수 있다.

Ⅴ. 결론 및 과제

보안 토큰 시스템은 안전한 사용자 인증의 요구를 만족시키기 위해 사용되며 사용자의 생체 정보를 보안 토큰상에서 인증함으로써 정보 유출이나 해킹에 대해 안전하다.

본 논문에서는 보안 토큰을 이용한 생체 인식 기술 동향을 살펴 보았고, USB를 이용한 보안 토큰 시스템을 실제로 설계 및 개발하고 시험한 내용을 기술하였다. 또한, 지문 정합 프로그램 수행 시에 수행 시간 및 메모리를 측정하고 고찰하였다.

최근 스마트 카드의 RAM 크기는 4~8kbytes 정도이므로 본 보안 토큰 시스템 상의 결과를 이용한 match-on-card 시스템에의 전이를 위해서는 수행 메모리 제약에 대한 연구가 필요하다. 또한, 스마트 카드는 카드 리더를 통한 호스트와의 통신에서 규정된 시간 제약성이 존재하므로 수행 시간에 대한 연구 또한 필요하다. 즉, 시간과 메모리에 대한 제약을 동시에 가지는 연구가 필요하다고 볼 수 있다.

현재, 지문 정합 프로그램에서 사용하는 RAM의 크기는 약 6.6kbytes이고 ROM 영역에 수용될 코드 크기는 약 43kbytes로 부팅 코드와 라이브러리의 자체 크기를 고려하더라도 제시한 보안 토큰 구현 스펙에 부합된다. 그러나 RAM 사용의 경우, 특징점 정보를 암복화하기 위해 암호 알고리듬을 사용하는 경우에는 키 및 중간 결과 데이터에 이용되는 수행 메모리 사용이 최고 수백 byte 이상이 요구될 것으로 예상된다. 따라서, match-on-token이 아닌, 메모리 제약이 더 심한 match-on-card 상에서는 특징점 저장 공간과 암호 알고리듬의 사용 공간에 대한 효율적인 메모리 공간 이용이 필요하다.

다중 생체 정보를 보안 토큰 시스템에 이용하는 내용도 앞으로의 연구과제이다. 현재 얼굴 정보 정합 프로그램을 이용한 보안 토큰 시스템을 구축중에 있는데, 이에 언급을 포함하여 지문 보안 토큰 시스템과의 성능 비교에 대한 내용 및 다중 생체 정보 보안 토큰 시스템 구축 등은 추후 결과에 포함될 예정이다.

참 고 문 헌

[1] 반성범 외, “사용자 인증을 위한 Match-on-Card 시스템에 관한 연구,” 제2회 생체인식기술 워크샵, 2002. 1.

[2] 장승석, Bulk transfer를 위한 USB host(Windows 98 PC) 프로그램 구성, 한국전자통신연구원 기술문서(TM-1700-1999-095), 1999. 10.

[3] Universal Serial Bus Specification 1.1, Sep. 23, 1998.

[4] S5N8946 ADSL/Cable Modem Microcontroller User’s Manual, Rev. 1.1, SAMSUNG, 2001.

[5] DaeSung Moon et al., “A Fingerprint Matching Algorithm Using A Multi-Resolution Scheme for Memory-Constrained Environments,” IC-AI’02, Vol. I, Las Vegas, 2002, pp. 124 - 128.

[6] http://www.arm.com.