

개봉 및 사용 전에 첨부된 제품 SLock 제품과 프로그램 사용계약서를 반드시 읽어주십시오.
이 소프트웨어를 구입,대여,개봉하여 사용하는 것은 ㈜대아미디어테크 제품,프로그램 사용계약서 내용에 동의함을 의미합니다.

SLock 제품과 소프트웨어 사용 계약서

모든 SLock제품들과 소프트웨어, 내용, 자료들에 대한 모든 Copyright는
㈜대아미디어테크 에서 갖고 있습니다.

SLock은 특허청에 1999년 상표로 등록되어 있습니다.

이 계약은 ㈜대아미디어테크와 프로그램 개발자, 사용자 사이에 법적인 사용허가계약으로 사용자가 구입하여 아래 계약 내용에 동의하지 않으면, 사용하지않은 모든 제품, 소프트웨어, 자료, 매뉴얼 등 일체를 파손 없이 구입한 날로부터 3일 이내에 돌려 주시면 기타 부대비용을 당사가 부담하지 않고, 제비용 공제 후 환불하여 드리고 구입계약을 해지 합니다.

1.사용권 : ㈜대아미디어테크 에서는 제품에 대한 사용권한을 드립니다.

SLock 하드웨어 제품의 케이스를 열거나, 부품의 수정, 변조, 파괴하면 사후 서비스가 불가능 합니다.

제품사용 시 제품을 떨어트리거나, 제품에 충격을 주어서는 안됩니다, 또한 안전한 보안을 위하여 제품을 컴퓨터에 설치 후 컴퓨터 프린터 포트 또는 USB 포트에서 분리하지 마십시오

SLock의 소프트웨어를 수정, 디스어셈블, decompile, reverse engineer 변조등으로 소프트웨어의 소스코드를 찾아내려 해서는 안됩니다.

제품의 모든 부속물에 대한 저작권과 지적 소유권은 ㈜대아미디어테크에서 가지고 있으며 이 권리는 대한민국의 저작권법과 국제저작권 조약에 의해 보호됩니다.

2.소프트웨어의 수정 : 컴퓨터에서 사용하는 소프트웨어의 불법복제를 방지할 목적으로 제공된 라이브러리를 사용, 결합, 링크할 수 있습니다.

3.보증의 한계(Limited Warranty)및 책임 : 제품은 구입일로 부터 90일 동안 SLock Key의 제작상 제품의 에러에 대해서만 교환하여 드리고, SLock 키 제품의 개봉, 수정, 파손, 파괴, 취급부주의 실수, 취급 소홀인 경우 교환이 불가능합니다.

제조사, ㈜대아미디어테크, 판매자는 이 제품에 포함된 모든 기능이 고객의 특정목적에 적합할 것이라는 보증은 하지 않으며, 본 제품의 사용으로 인해 초래된 모든 결과와 직.간접적인 손실, 손해에 대한 모든 책임을 지지 않습니다.

4.고객지원 : 고객의 제품사용을 위해 최선을 다한다.

5.인정 : 귀하는 이 사용계약서에 명시된 모든 문서내용을 읽고 이해하며, 계약조건에 동의하고, 나아가 이내용이 과거의 모든 약속, 주문, 광고, 고지 또는 서면합의 사항에 우선하는 것임을 인정합니다.

6.문의:문의사항이 있으시면 ㈜대아미디어테크 로 연락 주십시오

<http://www.slock.co.kr> 전화:(02)2634-4485

SLock 사용자 설명서 V10.6

21세기 정보화 사회로의 힘찬 도약을 준비하고 있는 지금,
그러나 우리는 아직도, 여전히 소프트웨어를 불법으로 복제, 유통시키고 있습니다.

이러한 환경 하에서의 소프트웨어 개발자들은
불법 복제에 따른 손실액을 제품 단가를 통해 회복하려 하고 있으며,
이는 곧바로 정식 사용자층의 부담으로 작용하여 제품 판매량을 저하시키고,
수익 감소에 따른 도산의 위기 속에서 업체들은,
제품 업그레이드는 접어 두고, 제품 포기 및 기술의 양도라는 최악의 상황도 직면하고 있습니다.
이러한 악순환으로 인해 국내 소프트웨어 산업은
질적, 양적 측면에서 급격한 하향세를 나타내는 반면,
복제 방지 장치로 무장한 외국 제품들은
자국내에서 축적한 자본을 토대로 꾸준한 업그레이드와 가격 인하를 무기로
국내 시장에 판매하고 있습니다.

SLock 은
소프트웨어의 불법 복제로 인해 야기되는 악순환의 고리를 끊고,
국내 소프트웨어 산업의 부흥과
지적 재산권의 보호 원칙이 통용되는 국민 정서 함양을 통해
진정한 정보화 사회로의 도약을 위해 일익을 담당하고자,
순수 국내 기술로 개발된 복제방지장치입니다.
개발 소프트웨어와 SLock 접목에 의한 이윤 증대로
개발자들은 제품의 질적 향상과 경제적인 가격 형성에 박차를 가할 수 있으며,
사용자들에게 저렴한 가격으로 보다 다양한 제품을 공급해 나갈 수 있을 것입니다.

목 차

SLock 소프트웨어 보안시스템

● 제품소개-----	4
● SLock 제품의 종류별 특성-----	6
● USB SLock 하드웨어 설치-----	7
- WIN XP 설치-----	7
- WIN7, Vista 32Bit 설치-----	12
- WIN7, Vista 64Bit 설치-----	15
- WIN98 설치-----	19
- WIN2000 설치-----	20
● SLock 제품 테스트-----	22
● SLock Editor-----	24
● SLock API-----	26
● SLock 오류코드-----	29
● SLock 예제-----	30

● 개요

SLock 소프트웨어보안시스템 은 많은 막대한 비용과 노력으로 개발된 소프트웨어 제품이 불법으로 복제, 유통되는 것을 차단할 할 수 있는 복제방지장치로서 하드웨어와 소프트웨어의 혼용으로 보안 작업을 할 수 있는 제품입니다. SLock제품은 단순 메모리 방식인 타제품에 비해, 마이크로칩 방식의 칩 설계와 첨단 알고리즘을 기반으로 메모리 영역 및, 데이터 입출력에 있어 보안성을 극대화하였으므로, 해킹 의도를 원천적으로 봉쇄하는 최신의 소프트웨어 보안 시스템입니다.

● 동작 방식

각 제품은 고유한 접근 코드를 가지며, 프로텍트된 프로그램은 실행시 특정한 접근 코드를 가진 SLock이 프린터 포트 또는 USB포트에 연결되어 있는지를 점검한 후, 접근 코드(PASSWORD)가 확인된 SLock 제품이 있으면, 소프트웨어가 실행되고, 그렇지 않을 경우에는 프로그램이 종료되거나, 테스트 용 버전으로 실행되는 등, .. 다양한 형태로 사용을 제한할 수 있습니다.

● SLock 개발자 키트 [Software Developer's Kit]

SLock SDK는 최초 고객들에게 제공되는 개발 테스트 버전 형식의 제품으로 원하시는 소프트웨어 제품에 SLock을 손쉽게 응용할 수 있도록 제공되어지고 있습니다.

④ 구성 요소

✓ SLock SDK 보안 개발 키트 는

평가 키트 SLock 하드웨어는 마스터 Password 12345, 3333 를 가지고 있으며 내부 메모리영역을 가지고 있는 제품입니다.

✓ 설치 드라이버 및 소프트웨어

SLock 장치 드라이버 설치 프로그램과 개발툴(언어)용 API 지원소프트웨어가 제공됩니다.

✓ DOS & WINDOWS 3.x API 소프트웨어 (SLock 페러럴용만 사용가능)

✓ WINDOWS 95/98, NT, XP API 소프트웨어 (SLock 페러럴용, USB 포트용 사용가능)

✓ WINDOWS Vista, WIN7 (32bit, 64bit) API 소프트웨어 (64bit 는 USB 포트용만 사용가능)

✓ SLock 사용자 설명서

● 프로그램 보호의 필요성

지금 이 시각에도

전 세계적으로 소프트웨어의 무단복제, 불법유통 및 사용은 계속되고 있습니다.

SPA [Software Publishers Association]와

BSA [Business Software Alliance]에 따르면

연간 소프트웨어 불법복제에 의한 손실액은 크며

계속적인 증가 추세라고 합니다.

불법복제는

소프트웨어 개발자들의 정신적, 물질적 피해뿐만 아니라,

개발 의지상실로 인해 국가의 소프트웨어 산업의 기반을 무너뜨리고,

사용자 또한 법적, 도덕적으로 자아상실을 초래하며,

더 나아가서 소프트웨어에 대한 외국의 종속을 초래할 수 있습니다.

● SLock 접목을 통한 개발자들의 장점

- ✓ 개발한 제품의 처음 판매시점부터 불법복제가 되지않아 판매가 증가 됩니다.
- ✓ 소프트웨어의 보호에 따른 이윤의 확대로
제품의 질적향상과 경제적인 가격형성에 박차를 가할 수 있으므로
사용자 층을 보다 확대할 수 있습니다.
- ✓ 실행 횟수나 사용날짜 제한등.. 다양한 형태로 프로젝트 작업에 적용할 수 있으므로,
제품 공급방식을 다각화할 수 있습니다.
- ✓ 사용자가 100% 등록이 가능하므로,
업그레이드 버전 및 새 상품에 대한 홍보 시 유용한 자료로 활용할 수 있습니다.

● SLock 접목을 통한 사용자들의 장점

- ✓ 원본 소프트웨어만을 이용하여야 하는 종래 방식의 불편함을 해소하여,
백업 또는 카피 등을 통해 프로그램 복구 시 보다 편리하게 사용할 수 있습니다.
- ✓ 정식등록을 통해 보다 다양한 서비스 혜택을 받으실 수 있습니다.
- ✓ 제품의 업그레이드 측면이나, 가격면에서 부가적인 혜택을 얻을 수 있습니다.
- ✓ 제품의 불법 유통이 근절됨으로, 불법복제에 따른 상대적 박탈감을 해소할 수 있습니다.
- ✓ 복수개의 복제장치 사용할 때 공간상으로나 분실의 염려가 있을 경우,
시스템상에 내장하여 사용할 수 있습니다.

◆ SLock 제품의 종류별 특성

SLock I,II	<p>첨단 보안 알고리즘과 마이크로 하드웨어 칩 설계를 통해 최적의 비용, 안전성과 신뢰성을 겸비한 제품입니다.</p> <p>추가적인 메모리 CMOS Floating 게이트 프로세서의 (2048bits-256자(8bit)) 기능을 통해 다방면에 응용할 수 있는 제품입니다.</p> <p>♠ 데모 버전 날짜제한, 대여 시 실행횟수 제한 등..</p> <p>♠ 유용한 정보를 저장하여 프로그램에서 호출하여 사용할 수 있습니다.</p> <p>♠ 모든 소프트웨어 사용자에게 독특한 코드를 할당할 수 있습니다.</p> <p>복수개의 복제방지장치를 사용할 경우, 공간상의 불편이나, 외관상의 문제점을 해소하고,추가적으로 USB 포트, 프린트 포트를 사용할 수 있습니다.</p>
USB SLock	<p>USB SLock 은 기존의 SLock(패러럴)의 기능을 가지고 있고 또한 메모리영역을 사용할 수 있습니다.</p> <p>기존의 프린터 포트용의 여러가지 프린터들의 사용으로 불편함과 충돌위험을 가지고 있으나 USB SLock은 컴퓨터에서 충돌 위험이 없고 컴퓨터는 많은 확장USB포트를 가지고 있어 사용하기에 편리합니다</p>

◆ 저희 SLock은 요 ?...

- ▶ 소량 주문하셔도 가능하므로 재고 부담을 해소합니다.
- ▶ 순수 국내 기술로 개발되어 신속한 납기와 업그레이드, A/S로 최선의 서비스를 해드립니다.
- ▶ 계속적으로 다량 구입하는 업체에 한하여, 주문자 방식의 설계 및, 다양한 색상의 제품을 공급하여 상품성을 높여 드립니다.

저렴하고, 안정적인 SLock 으로
귀하의 소중한 자산의 소프트웨어를 보호합니다.

최고의 소프트웨어는
최고의 소프트웨어 복제방지장치 SLock을 사용합니다.
훌륭한 소프트웨어는 국내 개발제품을 사용하여 국가 경제에 기여합니다.
1999년 SLock 상표 등록

PC(시스템)의 윈도우상에서 자동 지원하는 메모리 USB 드라이버 형식의 공개 HID 드라이버를 사용하는 것보다, 설치 시 불편하시더라도 보안시스템 전용의 개발전용 장치 드라이버를 사용하시는 것이 보안제품으로서 보안성이 뛰어납니다.

*****SLock 은 먼저 하드웨어 및 SLock 인식 전용 보안 드라이버를 설치 후, 개발툴(언어) 지원 API를 가지고 테스트 및 개발 프로그램에 작업을 할 수 있습니다 *****

♣ USB SLock 하드웨어 설치 (Win XP 사용 시)

* Windows XP 에서 USB SLock 드라이버 설치 방법

** Win XP 에서 USB SLock 드라이버 설치방법은 두가지의 방법을 사용할 수 있다. **

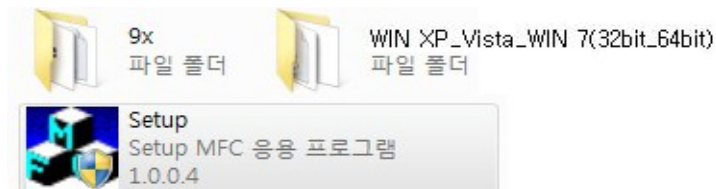
- A. USB SLock을 PC의 USB 포트에 연결 하지않고 설치하는 방법 또는
- B. USB SLock을 PC의 USB 포트에 연결 하고 설치하는 방법

1. USB SLock을 PC의 USB 포트에 연결 하지않고 Win XP 상에서 설치 방법

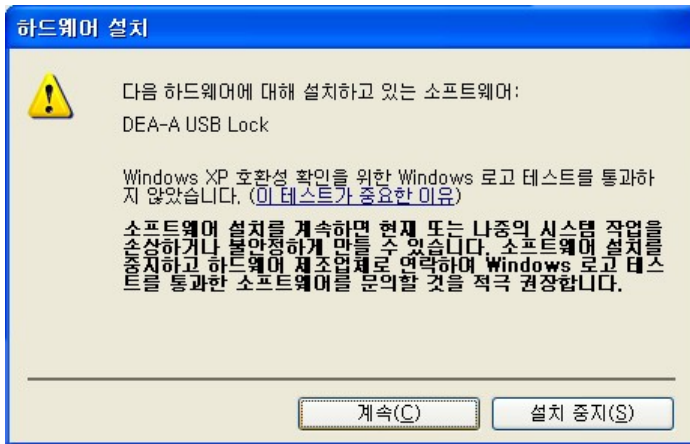
*** 먼저 USB SLock을 PC의 USB 포트에 연결하지 않고 드라이버를 설치 해야 한다. ***

제공된 USB SLock 드라이브 CD를 넣고 아래와 같이

설치 경로: CD-ROM WUSB_Driver\Setup.exe사용



[설치] 버튼 클릭.

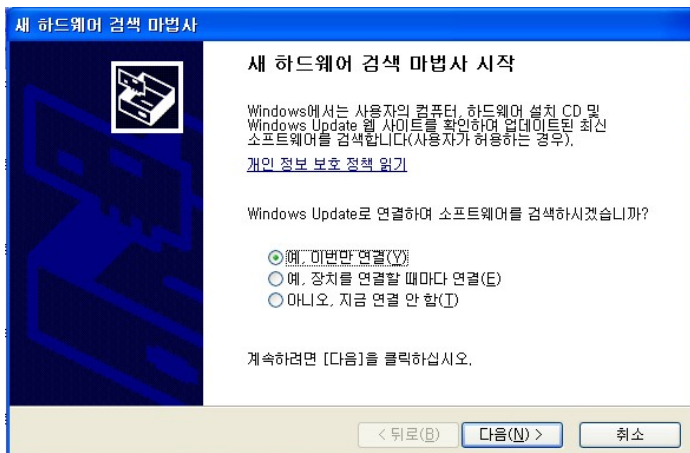


위의 창이 나타나면 [계속]을 클릭한다.

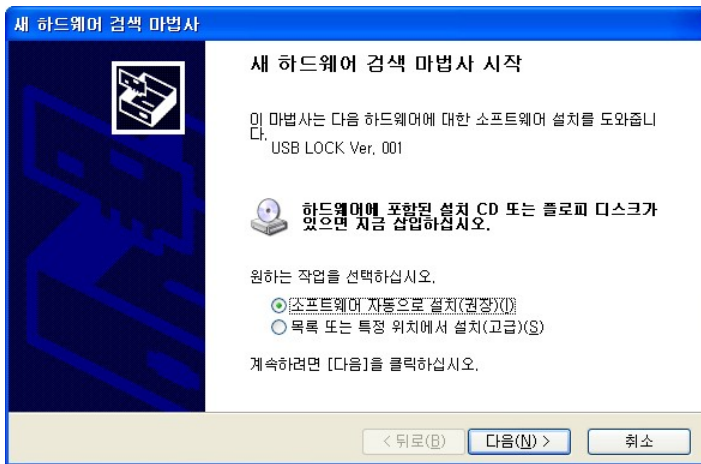


[확인]을 클릭한다.

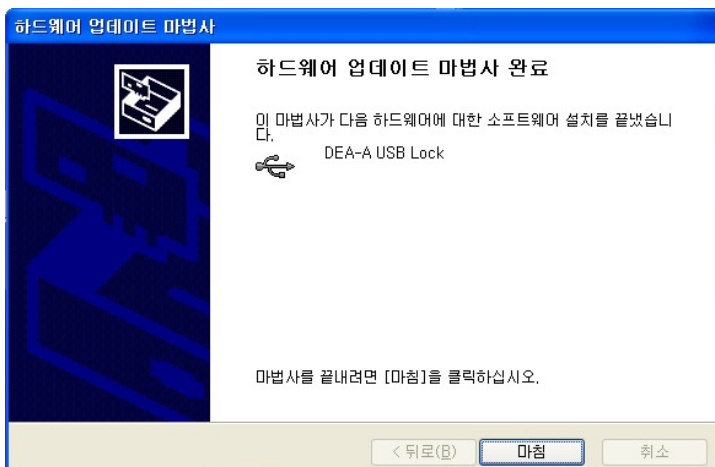
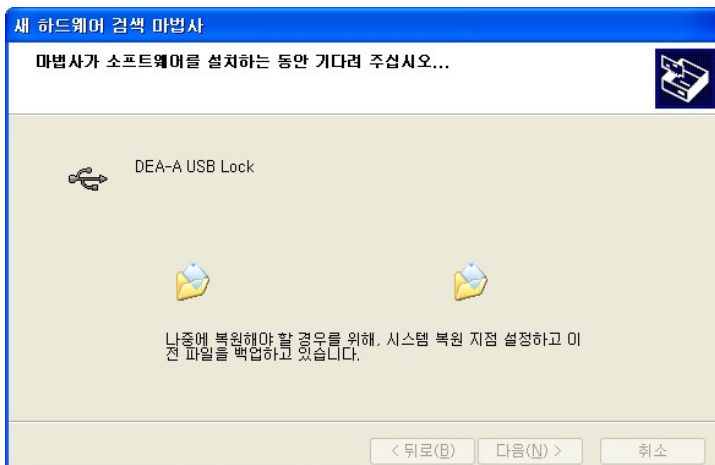
USB SLock을 PC의 USB 포트에 연결하면 다음과 같은 화면이 나타난다.



[예, 이번만 연결] 체크박스의 체크후 [다음]을 클릭한다.



[소프트웨어 자동으로 설치(권장)] 체크 후 [다음]을 클릭한다.



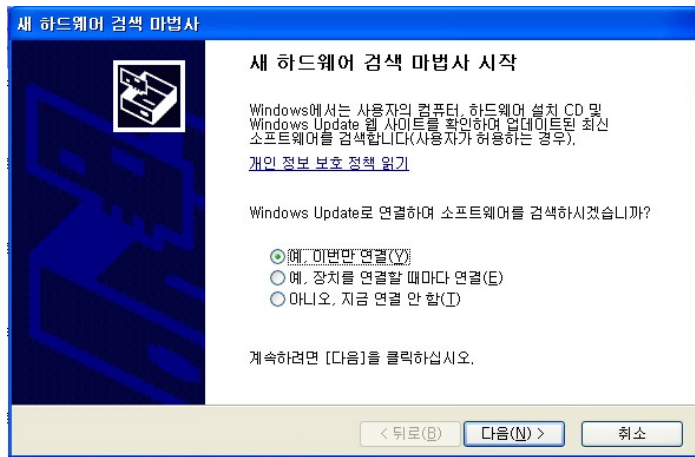
[마침]을 클릭 한다.

- USB SLock 드라이버 설치가 완료되었습니다. -
- 지금부터 사용 컴퓨터에서 보안시스템 전용의 장치 드라이버를 인식하는 것 입니다

2. USB SLock을 PC의 USB 포트에 연결하고 Win XP 설치 방법

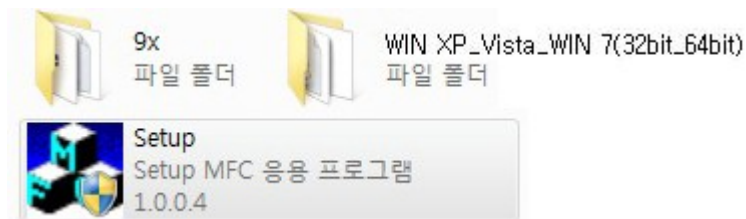
*** 먼저 USB SLock을 PC의 USB 포트에 연결하고 설치 해야 한다. ***

다음과 같은 창이 화면 우측 하단에 나타난다.



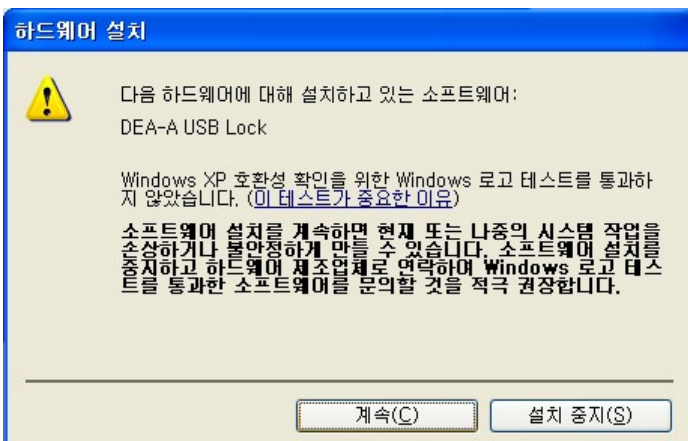
**** 위의 창이 나타나면 무시한다. ****

제공된 USB SLock 드라이브 CD를 넣고 아래와 같이
설치 경로: CD-ROM WUSB_Driver\Setup.exe사용





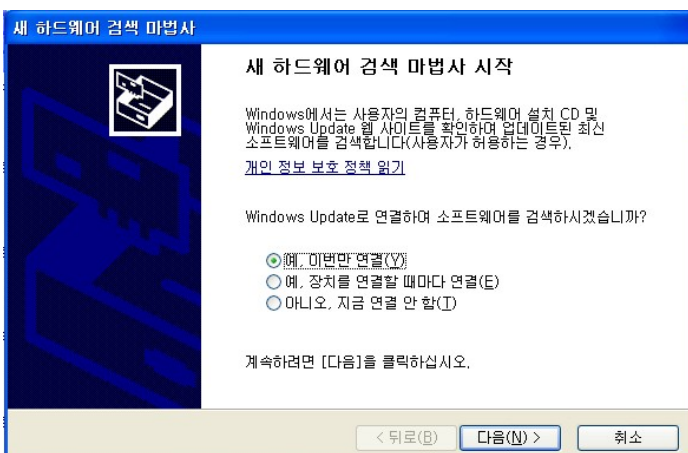
[설치] 버튼 클릭.



위의 창이 나타나면 [계속]을 클릭한다.



[확인]을 클릭한다.



위의 새 하드웨어 검색 마법사 창이 자동으로 사라지며 USB SLock드라이버가 설치된다.

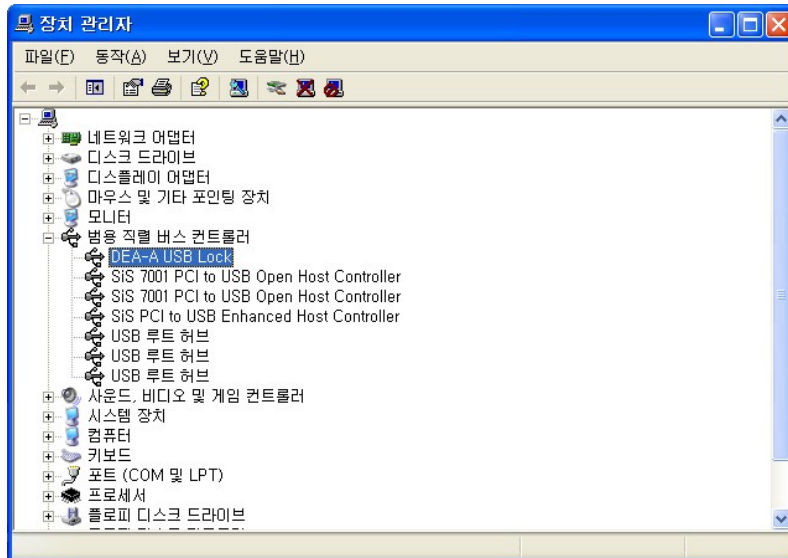
--- USB SLock 드라이버 설치가 완료 되었습니다.---

SLock 보안 장치 드라이버 파일 저장위치는 다음과 같다.

C:\WINDOWS\INF\SLockUsb.INF

C:\WINDOWS\System32\Drivers\SLOCKUSB.SYS

위와 같이 설치를 완료하고 장치관리자를 열어 드라이버가 정상적으로 설치 되어 있는지 확인한다.



장치관리자에서 USB SLock을 인식 하는지를 확인한다.

위의 표시된 부분은 정상적으로 USB SLock을 인식하는 화면이다.

(Vista, Win7 사용 시)

* Vista_x32, Win7_x32 의 USB SLock 드라이버 설치 방법 설명서

**** Vista_x32, Win7_x32 에서 USB SLock 드라이버 설치방법 ****

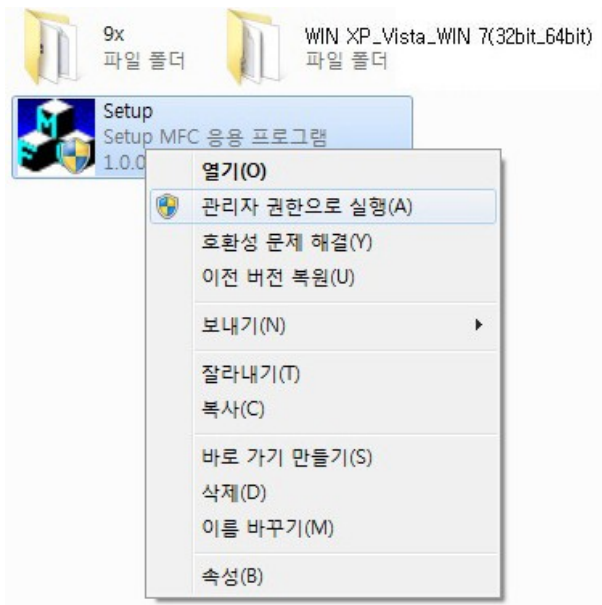
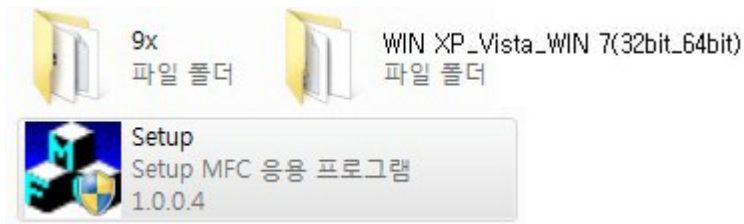
PC가 32비트 운영 체제를 확인하기 위해서는 위와 같이 **제어판 => 시스템 및 보안 => 시스템** 을클릭 하여 확인 합니다.

시스템 종류: 32비트 운영 체제 확인

***** USB SLock을 PC의 USB 포트에 연결하지 않고 드라이버를 설치 해야 한다. *****

제공된 USB SLock 드라이브 CD를 넣고 아래와 같이

설치 경로: CD-ROM\USB_Driver\Setup.exe사용



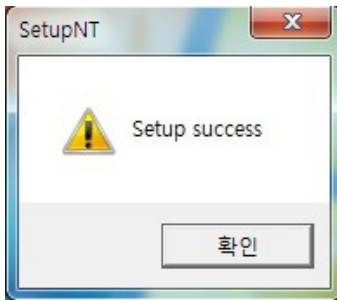
Setup을 오른쪽마우스를 실행하여 **관리자 권한으로 실행** 으로 클릭



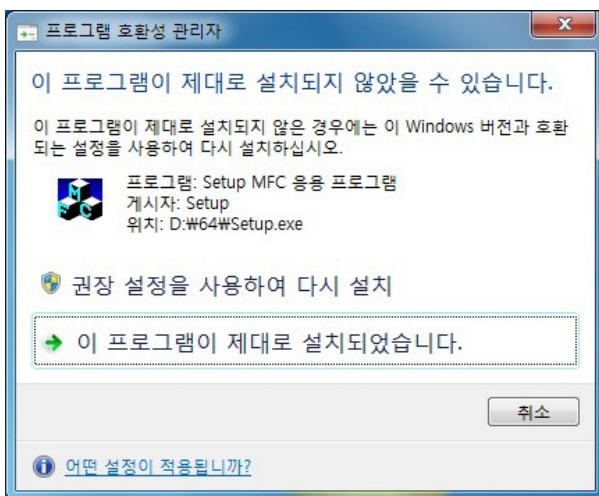
[설치]버튼 클릭



하단의 **이 드라이버 소프트웨어를 설치 합니다(I).** 클릭



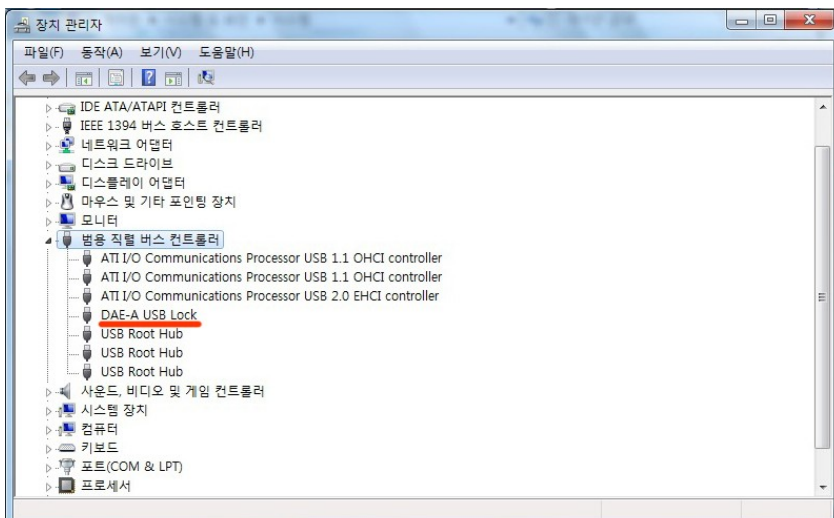
확인 클릭




이 프로그램이 제대로 설치되었습니다. 클릭

USB SLock 을 PC의 USB 포트에 연결합니다.

USB SLock 설치 확인은 제어판 => 시스템 및 보안 => 시스템 좌측 상단 장치 관리자 클릭



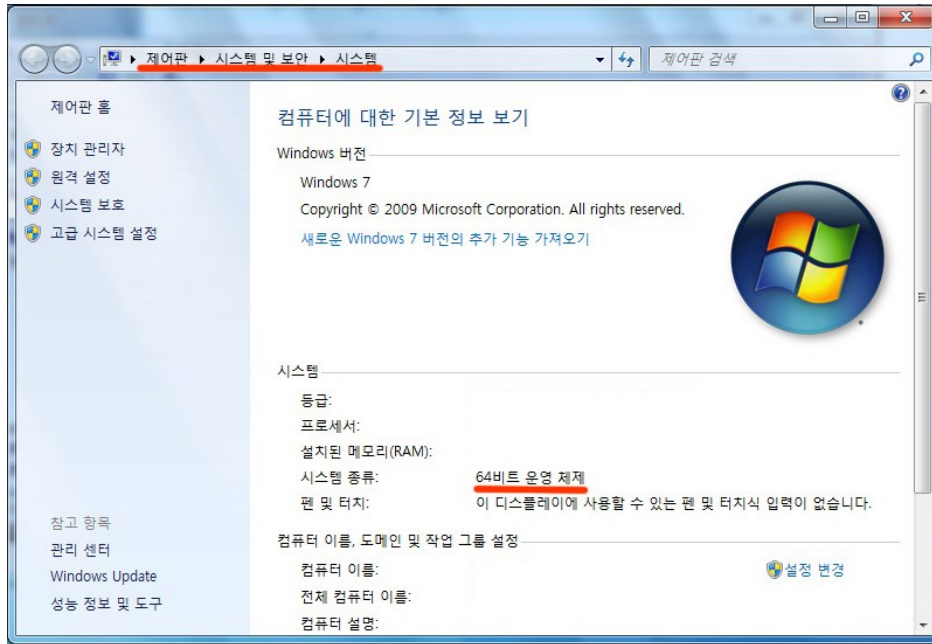
범용 직렬 버스 컨트롤러 클릭  DAE-A USB Lock 이라고 나타나면 정상 설치 완료 입니다.

64비트 컴퓨터에서 SLock 사용 시...

* Vista_x64, Win7_x64 의 USB SLock 드라이버 설치 방법 설명서

(Vista, Win7 사용 시...)

**** Vista_x64, Win7_x64 에서 USB SLock 드라이버 설치방법 ****



PC가 64비트 운영 체제를 확인하기 위해서는 위와 같이 **제어판 => 시스템 및 보안 => 시스템** 을클릭 하여 확인 합니다.

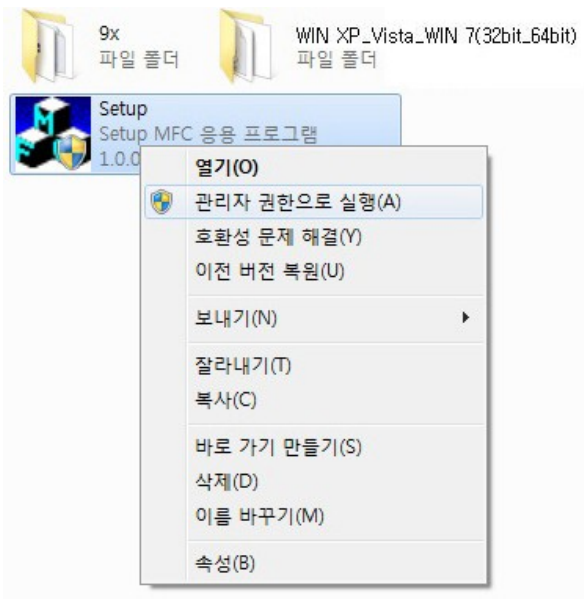
시스템 종류: 64비트 운영 체제 확인

***** USB SLock을 PC의 USB 포트에 연결하지 않고 드라이버를 설치 해야 한다. *****

제공된 USB SLock 드라이브 CD를 넣고 아래와 같이

설치 경로: CD-ROM WUSB_Driver\Setup.exe사용

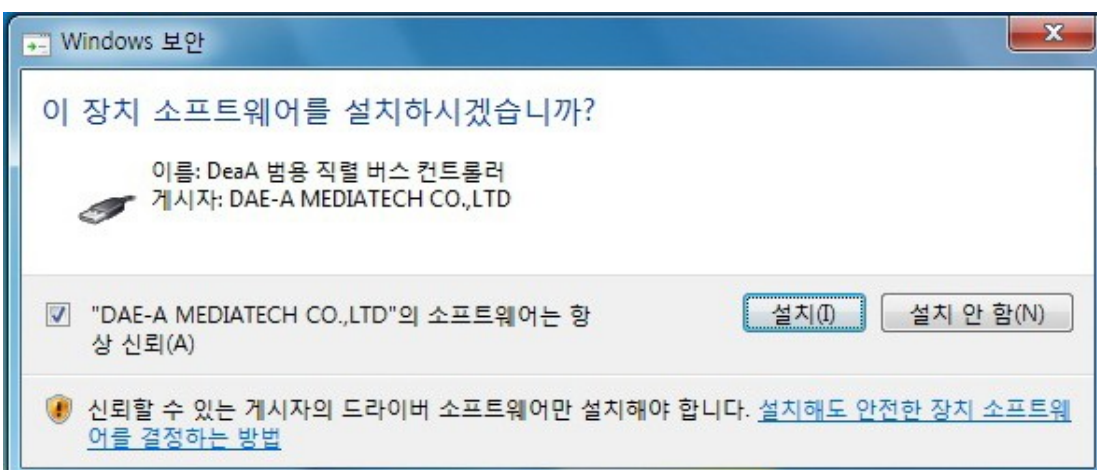




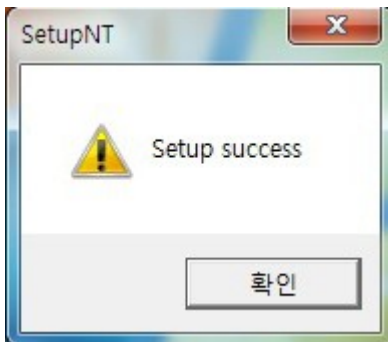
Setup을 오른쪽마우스를 실행하여 **관리자 권한으로 실행** 으로 클릭



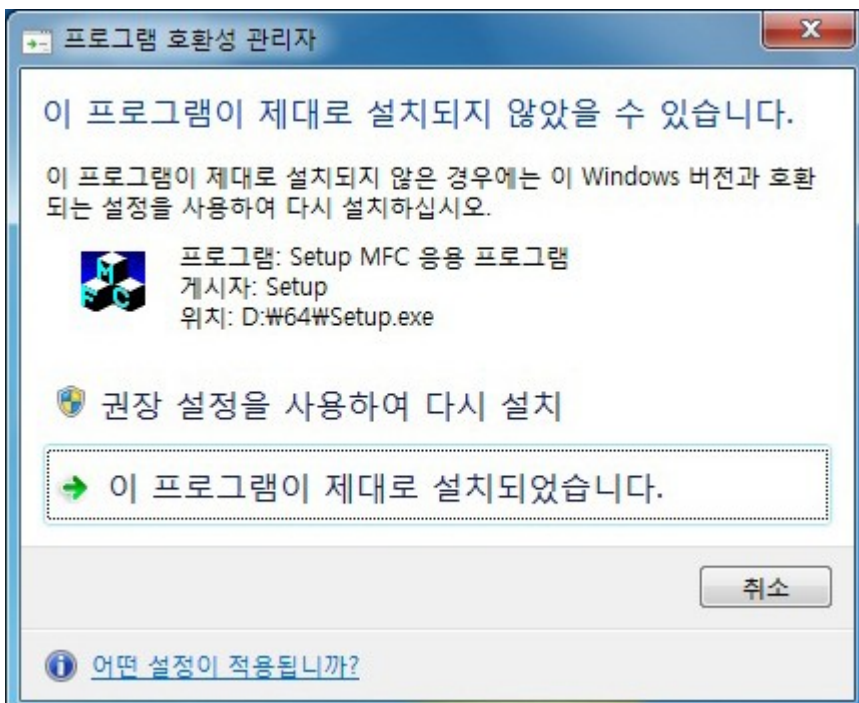
[**설치**]버튼 클릭



위와 같은 화면이 나타나면 좌측 ☒ **체크박스**에 체크 후 **설치** 버튼을 클릭



확인 클릭

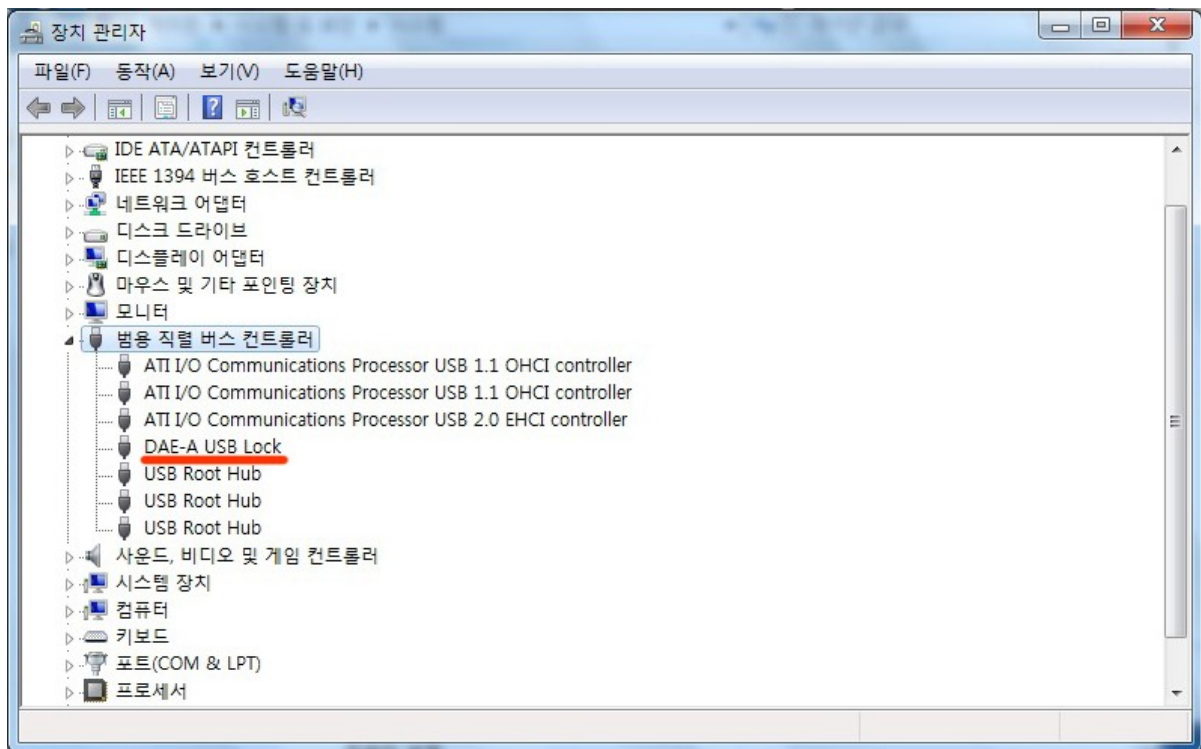



이 프로그램이 제대로 설치되었습니다. 클릭

USB SLock 을 PC의 USB 포트에 연결합니다.



설치 확인은 위와 같이 제어판 => 시스템 및 보안 => 시스템 좌측 상단 장치 관리자 클릭



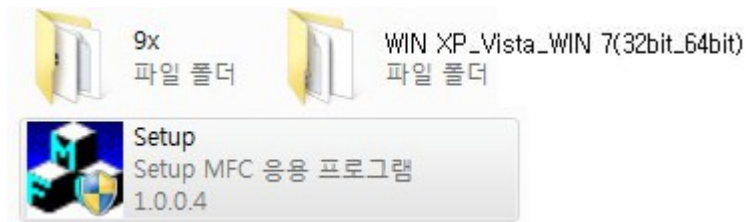
범용 직렬 버스 컨트롤러 클릭  DAE-A USB Lock 이라고 나타나면 정상 설치 완료 입니다.

* Windows 98 사용 시 USB SLock 드라이버 설치방법

*** USB SLock을 PC의 USB 포트에 연결하지 않고 드라이버를 설치 해야만 한다. ***

제공된 USB SLock 드라이브 CD를 넣고 아래와 같이

설치 경로: CD-ROM WUSB_Driver\Setup.exe사용



[설치] 버튼 클릭하면 창이 닫히며 드라이버 설치가 완료된다.

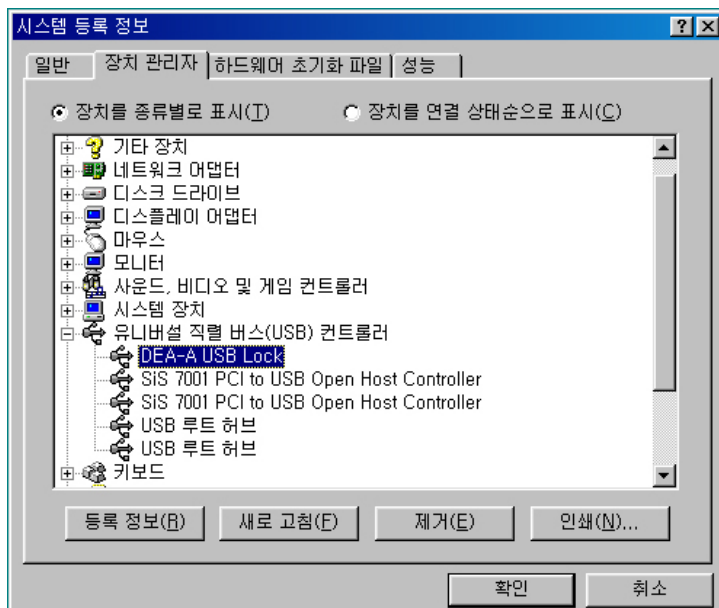
드라이버 파일 저장위치는 다음과 같다.

C:\WINDOWS\SYSTEM32\DRIVERS\SLOCKUSB.SYS

C:\WINDOWS\SYSTEM\WMM32.VXD(ntkern.vxd)

C:\WINDOWS\INF\SLOCKUSB.INF

위와 같이 설치를 완료하고 장치관리자를 열어 드라이버가 정상적으로 설치 되어 있는지 확인한다.



장치관리자에서 USB SLock을 인식 하는지를 확인한다.

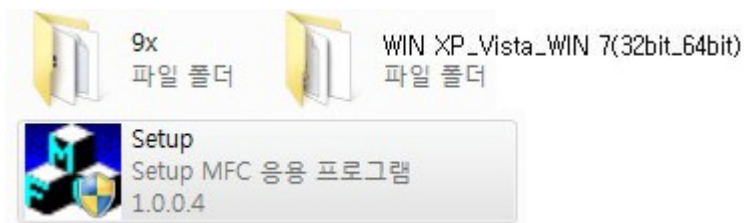
위의 표시된 부분은 정상적으로 USB SLock을 인식하는 화면이다.

* Windows 2000 사용 시 USB SLock 드라이버 설치방법

*** USB SLock을 PC의 USB 포트에 연결하지 않고 드라이버를 설치 해야만 한다. ***

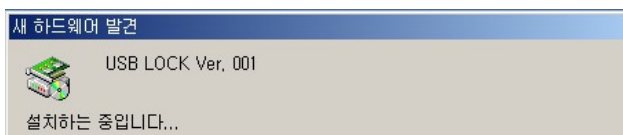
제공된 USB SLock 드라이브 CD를 넣고 아래와 같이

설치 경로: CD-ROM WUSB_Driver\Setup.exe사용



[설치] 버튼 클릭하면 창이 닫히며 드라이버 설치가 완료된다.

*** USB SLock을 PC의 USB 포트에 연결한다. ***



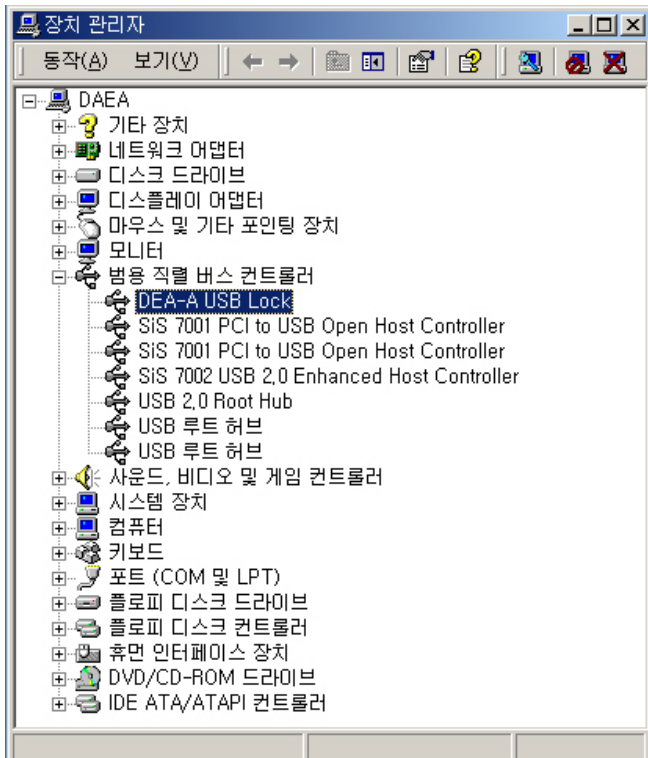
위의 화면이 나타나며 드라이버가 CD에서 자동 설치 된다.

드라이버 파일 저장위치는 다음과 같다.

C:\WINNT\WINFWSlockUsb.INF

C:\WINNT\System32\Drivers\WSLOCKUSB.SYS

위와 같이 설치를 완료하고 장치관리자를 열어 드라이버가 정상적으로 설치 되어 있는지 확인한다



장치관리자에서 USB SLock을 인식 하는지를 확인한다.

위의 표시된 부분은 정상적으로 USB SLock을 인식하는 화면이다.

▣ SLock 제품의 SDK (개발 키트) 테스트

1. SLock 하드웨어 및 드라이버 설치가 완료되면 개발자가 사용하는 개발툴(언어)별로 사용법을 확인(제공된 CDWTEST) 후 (사용은 헤더파일 이나 사용자가 각각의 USB SLock 체크를 원하는 곳에서 각 언어의 API를 적용하여 사용 한다)

USB SLock 은 각 개발툴(언어별) API를 제공하여 dll(C:\WINDOWS\system32에 저장되도록 하거나 개발하여 실행 하는 소프트웨어 폴더에서 사용)을 호출 하도록 프로그램 작업을 합니다.

2. USB SLock 사용법은 패스워드 체크 (고유번호 점검, 반환 값 점검, 번지 값 점검 및 변경)등으로 이루어져 있으며, **예를 들어** VC 를 이용하여 프로그램을 개발 하시는 분은 제공된 CDWTEST\VC32폴더에 예제 및 라이브러리 를 사용하여 프로텍션 작업을 하시면 됩니다

3. VC32폴더 안의 SLVC32.EXE 의 실행파일을 실행시(작업코드 1000)

정상적인 하드웨어 연결이 되었을 경우에는 하단에 “SLock II 가 정상적으로 연결되었습니다.” 라는 메시지가 나타나게 됩니다.

패스워드는 데모키(샘플제품)의 번호가 삽입 되어있습니다. (제품 구매시 제공하는 회사의 고유 패스워드를 삽입하여야 사용이 가능합니다.)

USB SLock 제품 구매시 PASSWORD는 업체별로 고유의 패스워드를 입력하여 드립니다.

4. 반환코드 점검(작업코드 1004)은 입력 값에 따라 반환 값은 달라질 수 있으며 암호와 입력 값이 동일할 경우 항상 동일한 반환 값을 얻을 수 있습니다.

5. 고유번호 점검(작업코드 1003)은 각각의 USB SLock 에 다른 번호가 입력되어 출고 됩니다.

6. 제품 버전 점검(작업코드 1002) 현재 USB SLock의 제품 버전 점검용 입니다.

7. 번지 값 점검(작업코드 1005) 부분은 USB SLock의 메모리 점검내용이며 Max memory가 119번지 까지 사용가능하며 주소값[0 - 119] 번지 중 몇 번지를 체크 할 것인가를 결정하는 것입니다. 예를 들어 10번지의 값을 알아보고 싶다면 10을 입력 하시면 10번지의 데이터가 출력이 되어 보여 집니다.

(아래의 그림은 미리 10번지의 데이터를 변환후 체크 한 것입니다.)

SLock II VC32 테스트 폼

패스 워드	12345	3333	반환 코드 점검
입력 코드	0	0	고유 번호 점검
반환 코드	53170	25541	제품 버전 점검
고유 번호	20173	26691	번지 값 점검
제품 버전	578	21333	번지 값 변환
주소값 [0 - 119]	10		종 료
데이터	USB : 1	3333	
상태 값	0	오류 번호	

8. 번지 값 변환(작업코드 1006)은 위와 같이 주소값과 데이터 값을 작성 후 번지 값 변환 버튼을 클릭 하면 다음과 같이 하단에 나타납니다.

SLock II VC32 테스트 폼

패스 워드	12345	3333	반환 코드 점검
입력 코드	0	0	고유 번호 점검
반환 코드	53170	25541	제품 버전 점검
고유 번호	20173	26691	번지 값 점검
제품 버전	578	21333	번지 값 변환
주소값 [0 - 119]	10		종 료
데이터	USB : 1	0	
상태 값	0	오류 번호	

지정 번지값의 데이터를 정상적으로 변경하였습니다. !!

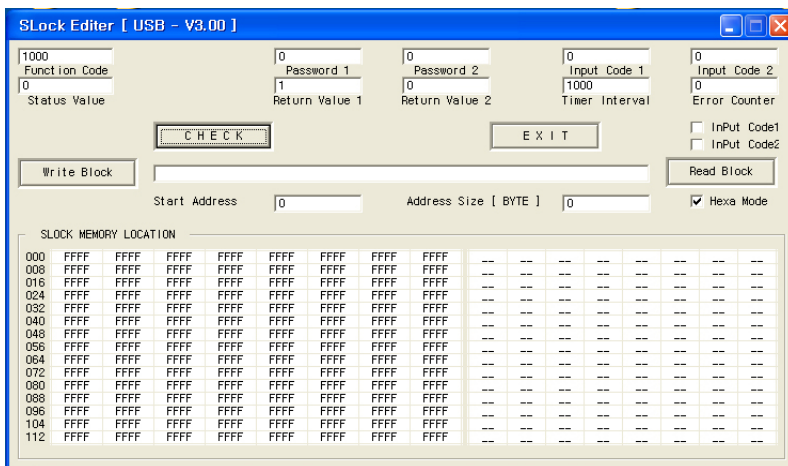
위의 작업코드 내용은 제공CD의 Slockusb_EditWSEdit.exe 에서도 동일하게 이용하실 수 있습니다.

☞ SLock Editor

파일 경로: CD-ROM\WSlockusb_Edit\WSEdit.exe

SLock Editor 는 Windows용 프로그램으로

SLock 상에서 메모리 영역의 데이터를 점검, 수정할 수 있도록 기본적으로 제공되는 프로그램입니다.



● 화면 구성

- ✓ HEXA 코드 상자 : 메모리 영역내에 데이터 값을 16 진수의 숫자 형태로 표시합니다.
- ✓ ASCII 코드 상자 : 메모리 영역내에 데이터 값을 ASCII 코드 형태로 표시합니다.
- ✓ Number 코드 라인 : 메모리 영역내에 데이터 값을 10 진수의 숫자 형태로 표시합니다.
SLock 오류 발생시 오류 코드를 표시합니다.
- ✓ Status 라인 : 커서 위치에 따른 메모리 영역이나, 작업 상태를 표시합니다.

SLock Editor 는 현재 시스템 상에

SLock이 정상적으로 장착이 되어 있는 경우에만 동작하므로 먼저 SLock를 연결하신 후에 Password 입력란에 현재 연결된 SLock의 접근 코드를 정확히 입력하시면 위의 화면이 나타납니다.

● SLock EDIT 메모리 영역 데이터 수정

SLock EDIT 프로그램은

최초 실행 시에 자동적으로 현재 메모리 영역내의 데이터를
각각 16 [HEXA] 진수, ASCII 코드, 10 [DEC]진수 형태로
화면상에 나타냅니다.

HEXA 코드, ASCII 코드 상자의 이동은 TAB 키를 이용하며,
선택된 상자는 황색으로 표시됩니다.

메모리 영역은 위의 그림에서와 같이
[0 - 119] 번지로 총 120 개 영역이 있으며,
각각 2 Byte 씩 사용이 가능하므로, 모두 240 Byte를 사용할 수 있습니다.
(프린터 포트용은 [0-104] 번지로 총 105개 영역 210 Byte 사용가능)

메모리 영역내의 위치를 이동하기 위해서 마우스를 이용하며,
현재 메모리 영역내의 데이터를 변경하기 위해서는
원하는 위치로 이동하여 25 마우스 더블 클릭 시 수정이 가능합니다.
물론 수정한 후에 메모리내의 저장 작업을 거쳐야만 수정 작업이 정상적으로 완료됩니다.

■ 작업 도중 SLock를 변경할 경우에는
프로그램을 종료한 후 다시 시작하여야 합니다.

◆ Function Code 기능

- ✓ 1002 - 현재 장착된 SLock의 버전을 확인할 경우에 사용합니다.
- ✓ 1003 - 현재 장착된 SLock의 고유 ID 번호를 확인할 경우에 사용합니다.
- ✓ 1004 - 현재 장착된 SLock의 특정값에 대한 반환 코드를 확인할 경우에 사용합니다.
- ✓ 1005 - 현재 장착된 SLock의 메모리 영역을 점검할 경우에 사용합니다.
- ✓ 1006 - 현재 장착된 SLock의 메모리 영역을 수정할 경우에 사용합니다.
- ✓ ESC - 프로그램을 종료할 경우에 사용합니다.

🔑 SLock API 선언

SLock은 다음의 “SLock” 26 함수를 통해 작업을 수행합니다.

SLock를 사용하기 위해서는

“SLock” 함수를 다음 형태로 선언한 후 원하시는 작업을 수행하시면 됩니다.

Status_Value sLock(Function_Code , PassWord1, PassWord2 , InPut_Code1, InPut_Code2 , *Return_Value1, *Return_Value2);		
Function_Code	작업 설정	[unsigned int]
PassWord1, PassWord2	설치된 SLock 접근 코드	[unsigned int]
InPut_Code1, InPut_Code2	작업시 필요한 데이터 입력	[unsigned int]
Return_Value1, Return_Value2	작업 수행 후 데이터 반환	[unsigned int]
Status_Value	정상적인 실행 여부 판단 값	[int]
26 “ Return_Value1, Return_Value2 “ 26 는 작업수행후 특정값이 기록됨으로 반드시 Value of Reference 형태로 선언되어야 합니다.		

♣ SLock 지원 작업 코드

작업	Function	Pass1	Pass2	InPut_1	InPut_2	Ret_1	Ret_2
OPEN_PORT	1000				✓	✓	
IS_SLOCK	1001					✓	
GET_VERSION	1002	✓	✓			✓	✓
GET_ID_NUM	1003	✓	✓			✓	✓
GET_CODE_NUM	1004	✓	✓	✓	✓	✓	✓
READ_WORD	1005	✓	✓	✓		✓	✓
WRITE_WORD	1006	✓	✓	✓	✓	✓	✓

☞ 기본 작업 및 메모리 관련작업 : SLock 모든 제품에서 사용 가능

OPEN_PORT, IS_SLOCK, GET_VERSION, GET_ID_NUM,

GET_CODE_NUM, READ_WORD, WRITE_WORD, ...

🔑 SLock API 사용법

OPEN_PORT	SLock 최초 실행시 반드시 우선적으로 사용여야 하는 명령으로 현재 SLock이 연결된 USB 포트 [USB] 를 감지하여, sLock 함수가 사용할 USB 포트를 체크하는 작업.
작업 코드	1000
입력 코드	InPut_Code2 : USB 포트 체크 [0 : 자동검색]
반환 코드	Return_Value1 : 작업중 오류 발생시 오류 코드 [Error Code] 반환
사용 예	Status_Value = sLock(1000, 0, 0, 0, 1, &Error_Code, 0);
참고 사항	OPEN_PORT 작업에서 sLock 함수가 사용할 USB 포트를 체크하는 작업입니다. InPut_Code2 값이 0 일 경우에는 USB포트에서 USB SLock을 자동 검색합니다.
주의 사항	두개 이상 복수의 USB SLock을 사용할 경우 SLock이 연결된 상위 포트가 먼저 인식됩니다.

IS_SLOCK	현재 시스템의 USB 포트 [OPEN_PORT 작업 실행 후 설정 값] 상에 SLock 이 정상적으로 연결되었는가를 확인하는 작업
작업 코드	1001
입력 코드	없음
반환 코드	Return_Value1 : 작업중 오류 발생시 오류 코드 [Error Code] 반환
사용 예	Status_Value = sLock(1001, 0, 0, 0, 0, &Error_Code, 0);

GET_VERSION	현재 설치된 SLock 의 제품 버전을 점검하는 작업
작업 코드	1002
입력 코드	PassWord1, PassWord2 : 지정된 SLock 의 암호 입력 [0 - 65535]
반환 코드	Return_Value1, Return_Value2 : 지정된 SLock 의 제품 버전 반환 Return_Value1 : 작업중 오류 발생시 오류 코드 [Error Code] 반환
사용 예	Status_Value = sLock(1002, 12345, 3333, 0, 0, &Ver1, &Ver2);
참고 사항	Status_Value 가 0 일 경우 Return_Value1 값이 1 이면, 현재 연결된 SLock 제품은 USB SLock 입니다.

GET_ID_NUM	현재 설치된 SLock 의 고유 번호를 점검하는 작업(ex: 고객관리용으로 활용가능)
작업 코드	1003
입력 코드	PassWord1, PassWord2 : 지정된 SLock 의 암호 입력 [0 - 65535]
반환 코드	Return_Value1, Return_Value2 : 지정된 SLock 의 고유 번호 반환 Return_Value1 : 작업중 오류 발생시 오류 코드 [Error Code] 반환
사용 예	Status_Value = sLock(1003, 12345, 3333, 0, 0, &ID1,&ID2);

GET_CODE	현재 설치된 SLock 의 반환 값을 점검하는 작업
작업 코드	1004
입력 코드	PassWord1, PassWord2 : 지정된 SLock 의 암호 입력 [0 - 65535] InPut_Code1 : 반환 값을 받기 위한 입력 값 [0 - 65535] InPut_Code2 : 반환 값을 받기 위한 입력 값 [0 - 65535]
반환 코드	Return_Value1, Return_Value2 : SLock 의 입력 값에 따른 반환 값 Return_Value1 : 작업중 오류 발생시 오류 코드 [Error Code] 반환
사용 예	Status_Value = sLock(1004, 12345, 3333, 0, 10, &Code1, &Code2);
참고 사항	InPut_Code1,2 에 의한 입력 값에 따라 반환 값은 달라질 수 있으며, 암호와 입력 값이 동일할 경우에는 항상 동일한 반환 값을 얻을 수 있습니다.

READ_WORD	현재 설치된 SLock 의 메모리 번지 내의 데이터를 점검하는 작업
작업 코드	1005
입력 코드	PassWord1, PassWord2 : 지정된 SLock 의 암호 입력 [0 - 65535] InPut_Code1 : 데이터 점검을 원하는 메모리 번지 값 입력 [0 - 119]
반환 코드	Return_Value1 : 지정된 메모리 번지 내의 데이터 반환 값 Return_Value1 : 작업중 오류 발생시 오류 코드 [Error Code] 반환
사용 예	Status_Value = sLock(1005, 12345, 3333, 20, 0, &Data, 0);

WRITE_WORD	현재 설치된 SLock 메모리 번지 내의 데이터를 변경하는 작업
작업 코드	1006
입력 코드	PassWord1, PassWord2 : 지정된 SLock 의 암호 입력 [0 - 65535] InPut_Code1 : 데이터 점검을 원하는 메모리 번지 값 입력 [0 - 119] InPut_Code2 : 지정된 메모리 번지 내의 변경할 데이터 지정
반환 코드	Return_Value1 : 작업중 오류 발생시 오류 코드 [Error_Code] 반환]
사용 예	Status_Value = sLock(1006, 12345, 3333, 20, 100, &ErrorCode, 0);

🔍 오류 코드 : Status_Value 값이 1 일 경우에 발생

✓ 코드 번호 - 1000

현재 시스템 [] 상에

SLock 장치 드라이버 [SlockUsb.sys, SlockUsb64.Sys ..]가

정상적으로 설치되지 않았을 경우에 발생합니다.

✓ 코드 번호 - 1001

OPEN_PORT [1000] 작업시

USB포트 상에 SLock 이 연결되지 않았을 경우에 발생합니다.

✓ 코드 번호 - 1002

현재 시스템에 SLock 이 연결되지 않아

정상적인 데이터를 반환 받지 못할 경우에 발생합니다.

✓ 코드 번호 - 1003

현재 SLock 작업 실행 중 비정상적인 값을 입력할 경우에 발생합니다.

✓ 코드 번호 - 1004

현재 시스템에 연결된 SLock과 입력된

접근 코드가 동일하지 않을 경우에 발생합니다.

✓ 코드 번호 - 1005

현재 소프트웨어 버전상에서는

지원되지 않는 SLock 작업 코드를 사용할 경우에 발생합니다.

✓ 코드 번호 - 1006

현재 시스템에 연결된 SLock 제품이

지원하지 않는 작업 코드를 사용할 경우에 발생합니다.

▶ 다음은 SLock 을 VC++ 언어에서 사용할 경우로
제공 라이브러리에 포함된 간단한 예제입니다.

파일 경로: CD-ROM -> Test -> VC32 -> Slock.h

```
#include "RESOURCE.H"
```

```
// Function Code (SLock 작업코드)
```

```
#define OPEN_PORT          1000
```

```
#define IS_SLOCK           1001
```

```
#define GET_VERSION        1002
```

```
#define GET_ID_NUM         1003
```

```
#define GET_CODE_NUM       1004
```

```
#define READ_WORD          1005
```

```
#define WRITE_WORD         1006
```

```
// Status Code
```

```
#define SLOCK_OK           0
```

```
#define SLOCK_NOT_OK       1
```

```
#define Max_Memory         119
```

```
#ifdef __cplusplus
```

```
extern "C" {
```

```
#endif
```

```
__declspec(dllimport) int CALLBACK sLock(UINT FuncCode, UINT Pass1, UINT Pass2, UINT  
InPut1, UINT InPut2, UINT *Retrun1, UINT *Return2);
```

```
#ifdef __cplusplus
```

```
};
```

```
#endif
```

```
LRESULT CALLBACK sLock_Test(HWND, UINT, WPARAM, LPARAM);
```

```
void    Check_Multi_Port(HWND);
```

```
UINT    Current_PassWord1 = 12345, Current_PassWord2 = 3333;
```

```
UINT    Old_PassWord1 = 0, Old_PassWord2 = 0;
```

```
unsigned char Temp[] = "                                ";
```

```
LRESULT CALLBACK sLock_Test(HWND hDlg, UINT Msg, WPARAM wParam, LPARAM lParam)
```

```
{
```

```
    int Status_Value;
```

```
    unsigned int Return_Value1, Return_Value2;
```

```
    switch (Msg) {
```

```
        case WM_INITDIALOG :
```

```
            SetDlgItemInt(hDlg, IDC_Pass1_Box, 12345, 0);
```

```
            SetDlgItemInt(hDlg, IDC_Pass2_Box, 3333, 0);
```

```
            SetDlgItemInt(hDlg, IDC_InPut1_Box, 0, 0);
```

```
            SetDlgItemInt(hDlg, IDC_InPut2_Box, 0, 0);
```

```
            SetDlgItemInt(hDlg, IDC_Address_Box, 0, 0);
```

```
            SetDlgItemInt(hDlg, IDC_Data_Box, 0, 0);
```

```

// OPEN_PORT [ 1000 ] - SLock 최초 실행시 사용하는 명령으로
//             현재 SLock이 연결된 포트를 자동으로 감지하여,
//             아래 함수들이 사용할 포트를 지정하는 작업.
// sLock(OPEN_PORT, 0, 0, 0, Port_Num, &Return_Value1, 0)
// Port_Num - 0 : 자동 검색 ...
    Status_Value = sLock(OPEN_PORT, 0, 0, 0, 0, &Return_Value1, 0);
    Check_Multi_Port(hDlg);
// IS_SLOCK [ 1001 ] - 현재 시스템의 포트[ OPEN_PORT 작업 실행 후 설정 값 ] 상에
//             SLock 이 정상적으로 연결되었는가를 확인하는 작업
//             sLock(IS_LOCK, 0, 0, 0, 0, &Return_Value1, 0)
    Status_Value = sLock(IS_SLOCK, 0, 0, 0, 0, &Return_Value1, 0);
    SetDlgItemInt(hDlg, IDC_Status_Value_Box, Status_Value, 0);

    if (Status_Value == SLOCK_OK) {

        SetDlgItemText(hDlg, IDC_Status_Box, " SLock이 정상적으로 연결되었습니다.");

        SetDlgItemText(hDlg, IDC_Error_Box, "");

    }

    else {
        SetDlgItemText(hDlg, IDC_Status_Box, "오류 발생 !! ");
        SetDlgItemInt(hDlg, IDC_Error_Box, Return_Value1, 0);

    }

    break;

case WM_COMMAND :

    switch (LOWORD (wParam)) {

        case ID_Exit :

            EndDialog(hDlg, IDOK);

            break;

```


// GET_ID_NUM [1003] - 현재 설치된 SLock의 고유 번호를 점검하는 작업

case IDC_ID_Num_Test :

SetDlgItemText(hDlg, IDC_Status_Box, Temp);

SetDlgItemText(hDlg, IDC_Error_Box, "");

SetDlgItemInt(hDlg, IDC_ID_Num1_Box, 0, 0);

SetDlgItemInt(hDlg, IDC_ID_Num2_Box, 0, 0);

// sLock(GET_ID_NUM, Pass1, Pass2, 0, 0, ID_Num1, ID_Num2)

Status_Value = sLock(GET_ID_NUM,

GetDlgItemInt(hDlg, IDC_Pass1_Box, NULL, 0),

GetDlgItemInt(hDlg, IDC_Pass2_Box, NULL, 0),

0, 0,

&Return_Value1, &Return_Value2);

if (Status_Value == SLOCK_OK) {

SetDlgItemInt(hDlg, IDC_ID_Num1_Box, Return_Value1, 0);

SetDlgItemInt(hDlg, IDC_ID_Num2_Box, Return_Value2, 0);

}

else {

SetDlgItemText(hDlg, IDC_Status_Box, " 오류 발생 !! ");

SetDlgItemInt(hDlg, IDC_Error_Box, Return_Value1, 0);

}

SetDlgItemInt(hDlg, IDC_Status_Value_Box, Status_Value, 0);

break;

// GET_VERSION [1002] - 현재 설치된 SLock의 제품 버전을 점검하는 작업

case IDC_Version_Test :

SetDlgItemText(hDlg, IDC_Status_Box, Temp);

SetDlgItemText(hDlg, IDC_Error_Box, "");

SetDlgItemInt(hDlg, IDC_Version1_Box, 0, 0);

SetDlgItemInt(hDlg, IDC_Version2_Box, 0, 0);

// sLock(GET_VERSION, Pass1, Pass2, 0, 0, Version1, Version2)

Status_Value = sLock(GET_VERSION,

GetDlgItemInt(hDlg, IDC_Pass1_Box, NULL, 0),

GetDlgItemInt(hDlg, IDC_Pass2_Box, NULL, 0),

0, 0,

&Return_Value1, &Return_Value2);

if (Status_Value == SLOCK_OK) {

SetDlgItemInt(hDlg, IDC_Version1_Box, Return_Value1, 0);

SetDlgItemInt(hDlg, IDC_Version2_Box, Return_Value2, 0);

}

else {

SetDlgItemText(hDlg, IDC_Status_Box, " 오류 발생 !! ");

SetDlgItemInt(hDlg, IDC_Error_Box, Return_Value1, 0);

}

SetDlgItemInt(hDlg, IDC_Status_Value_Box, Status_Value, 0);

break;

// GET_CODE_NUM [1004] - 현재 설치된 SLock의 반환 값을 점검하는 작업

case IDC_Code_Num_Test :

SetDlgItemText(hDlg, IDC_Status_Box, Temp);

SetDlgItemText(hDlg, IDC_Error_Box, "");

SetDlgItemInt(hDlg, IDC_Code_Num1_Box, 0, 0);

SetDlgItemInt(hDlg, IDC_Code_Num2_Box, 0, 0);

if (GetDlgItemInt(hDlg, IDC_InPut1_Box, NULL, 0) <= 0

|| GetDlgItemInt(hDlg, IDC_InPut1_Box, NULL, 0) > 65535)

SetDlgItemInt(hDlg, IDC_InPut1_Box, 0, 0);

if (GetDlgItemInt(hDlg, IDC_InPut2_Box, NULL, 0) <= 0

|| GetDlgItemInt(hDlg, IDC_InPut2_Box, NULL, 0) > 65535)

SetDlgItemInt(hDlg, IDC_InPut2_Box, 0, 0);

SetDlgItemInt(hDlg, IDC_InPut1_Box, GetDlgItemInt(hDlg, IDC_InPut1_Box, NULL, 0), 0);

SetDlgItemInt(hDlg, IDC_InPut2_Box, GetDlgItemInt(hDlg, IDC_InPut2_Box, NULL, 0), 0);

// sLock(GET_CODE_NUM, Pass1, Pass2, InPut1_Num, InPut2_Num, Code_Num1, Code_Num2)

Status_Value = sLock(GET_CODE_NUM,

GetDlgItemInt(hDlg, IDC_Pass1_Box, NULL, 0),

GetDlgItemInt(hDlg, IDC_Pass2_Box, NULL, 0),

GetDlgItemInt(hDlg, IDC_InPut1_Box, NULL, 0),

GetDlgItemInt(hDlg, IDC_InPut2_Box, NULL, 0),

&Return_Value1, &Return_Value2);

if (Status_Value == SLOCK_OK) {

SetDlgItemInt(hDlg, IDC_Code_Num1_Box, Return_Value1, 0);

SetDlgItemInt(hDlg, IDC_Code_Num2_Box, Return_Value2, 0);

```

    }

else {

    SetDlgItemText(hDlg, IDC_Status_Box, " 오류 발생 !! ");

    SetDlgItemInt(hDlg, IDC_Error_Box, Return_Value1, 0);

    }

    SetDlgItemInt(hDlg, IDC_Status_Value_Box, Status_Value, 0);

    break;
// READ_WORD [ 1005 ] - 현재 설치된 SLock의 메모리 번지 내 데이터를 점검하는 작업
case IDC_Read_Test :
    SetDlgItemText(hDlg, IDC_Status_Box, Temp);
    SetDlgItemText(hDlg, IDC_Error_Box, "");
    SetDlgItemInt(hDlg, IDC_Data_Box, 0, 0);

    if (GetDlgItemInt(hDlg, IDC_Address_Box, NULL, 0) <= 0
        || GetDlgItemInt(hDlg, IDC_Address_Box, NULL, 0) > 65535)
        SetDlgItemInt(hDlg, IDC_Address_Box, 0, 0);
SetDlgItemInt(hDlg, IDC_Address_Box, GetDlgItemInt(hDlg, IDC_Address_Box, NULL, 0), 0);
// sLock(READ_WORD, Pass1, Pass2, Address, 0, Return_Word, 0)
Status_Value = sLock(READ_WORD,
    GetDlgItemInt(hDlg, IDC_Pass1_Box, NULL, 0),
    GetDlgItemInt(hDlg, IDC_Pass2_Box, NULL, 0),
    GetDlgItemInt(hDlg, IDC_Address_Box, NULL, 0),
    0,
    &Return_Value1, 0);
if (Status_Value == SLOCK_OK) {
    SetDlgItemInt(hDlg, IDC_Data_Box, Return_Value1, 0);
}

```

```
else {
```

```
    SetDlgItemText(hDlg, IDC_Status_Box, " 오류 발생 !! ");
```

```
    SetDlgItemInt(hDlg, IDC_Error_Box, Return_Value1, 0);
```

```
}
```

```
    SetDlgItemInt(hDlg, IDC_Status_Value_Box, Status_Value, 0);
```

```
    break;
```

```
// WRITE_WORD [ 1006 ] - 현재 설치된 SLock의 메모리 번지 내 데이터를 변경하는 작업
```

```
case IDC_Write_Test :
```

```
    SetDlgItemText(hDlg, IDC_Status_Box, Temp);
```

```
if (GetDlgItemInt(hDlg, IDC_Address_Box, NULL, 0) <= 0
```

```
    || GetDlgItemInt(hDlg, IDC_Address_Box, NULL, 0) > 65535)
```

```
    SetDlgItemInt(hDlg, IDC_Address_Box, 0, 0);
```

```
if (GetDlgItemInt(hDlg, IDC_Data_Box, NULL, 0) <= 0
```

```
    || GetDlgItemInt(hDlg, IDC_Data_Box, NULL, 0) > 65535)
```

```
    SetDlgItemInt(hDlg, IDC_Data_Box, 0, 0);
```

```
    SetDlgItemInt(hDlg, IDC_Address_Box, GetDlgItemInt(hDlg, IDC_Address_Box, NULL, 0), 0);
```

```
// sLock(WRITE_WORD, Pass1, Pass2, Address, InPut_Data, Error_Code, 0)
```

```
Status_Value = sLock(WRITE_WORD,
```

```
    GetDlgItemInt(hDlg, IDC_Pass1_Box, NULL, 0),
```

```
    GetDlgItemInt(hDlg, IDC_Pass2_Box, NULL, 0),
```

```
    GetDlgItemInt(hDlg, IDC_Address_Box, NULL, 0),
```

```
    GetDlgItemInt(hDlg, IDC_Data_Box, NULL, 0),
```

```
    &Return_Value1, 0);
```

```
SetDlgItemInt(hDlg, IDC_Status_Value_Box, Status_Value, 0);
```

```
if (Status_Value == SLOCK_OK)
```

```
    SetDlgItemText(hDlg, IDC_Status_Box, " 지정 번지값의 데이터를 정상적으로 변경하였습니다. !! ");
```

```
else {
```

```
    SetDlgItemText(hDlg, IDC_Status_Box, " 오류 발생 !! ");
```

```
    SetDlgItemInt(hDlg, IDC_Error_Box, Return_Value1, 0);
```

```
}
```

```
SetDlgItemInt(hDlg, IDC_Status_Value_Box, Status_Value, 0);
```

```
SetDlgItemInt(hDlg, IDC_Data_Box, 0, 0);
```

```
break;
```

```
}
```

```
****감사 합니다****
```