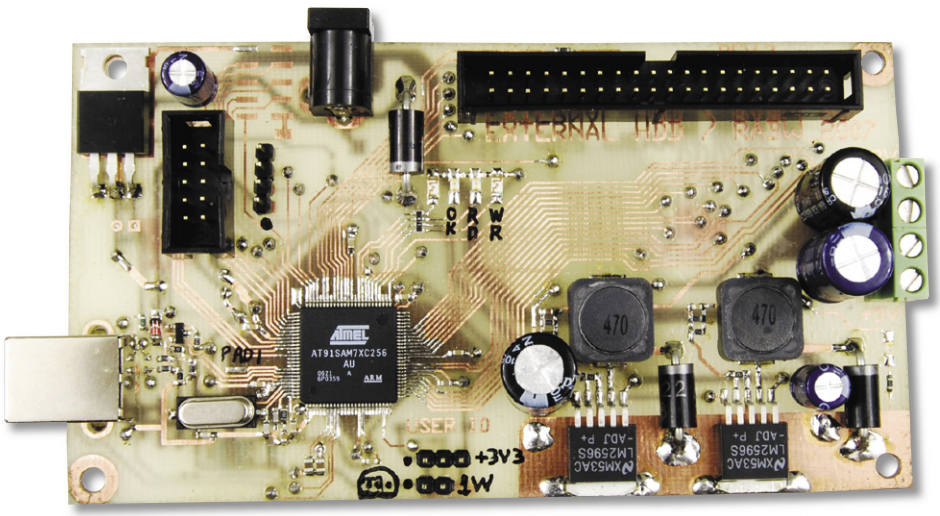
Bezpieczny dysk twardy

przejściówka USB–ATA z szyfrowaniem danych AES (2)

Ostatnio wiele się mówi a to o zaginionych laptopach polityków, a to o zniszczonych dyskach w ich komputerach. Jak pamiętamy, głośno było również o zatopieniu notebooka w wannie. Czy były to tylko przypadki, czy nieudolne próby zakamuflowania danych zapisanych na dyskach twardych?

Rekomendacje:
projekt będzie szczególnie przydatny dla tych, którzy mają coś do ukrycia na swoich komputerach.



Chroń dane na swoim dysku

AVT-5139

W ofercie AVT jest dostępna:
– [AVT–5139A] – płytka drukowana

PODSTAWOWE PARAMETRY

- Płytka o wymiarach 122,5x68 mm
- Zasilanie: 18 V z zewnętrznego zasilacza niestabilizowanego
- Obsługiwane dyski twarde: Parallel ATA, do 2 TB
- Interfejs: USB 2.0, Full Speed
- Szyfrowanie danych: AES, 128 bit
- Klasa urządzenia: Mass Storage Class (pamięć masowa USB), nie są wymagane dodatkowe sterowniki pod Windows 2k i XP
- Rzeczywista szybkość transmisji danych: około 0,4...0,5 MB/s
- Mikrokontroler: AT91SAM7XC256 (ARM7TDMI)
- Typowe zastosowania: zabezpieczenie prywatnych danych przed niepowołanym dostępem

PROJEKTY POKREWNE wymienione artykuły są w całości dostępne na CD

Tytuł artykułu	Nr EP/EdW	Kit
Konwerter USB-RS232	EP 9-10/2002	AVT-5080
Konwerter USB-IDE	EP 1/2003	AVT-5096
Konwerter USB-IrDA	EP 4/2003	AVT-510
Konwerter USB-RS232	EP 12/2003	AVT-556
Konwerter USB2.0-IDE	EP 5-6/2005	AVT-387
Konwerter USB na RS232 z układem FT232R	EP 12/2005	–
Przejściówka USB-LPT	EP 4/2007	AVT-981
Konwerter USB-IO	EP 7/2008	AVT-5140

Montaż i uruchomienie

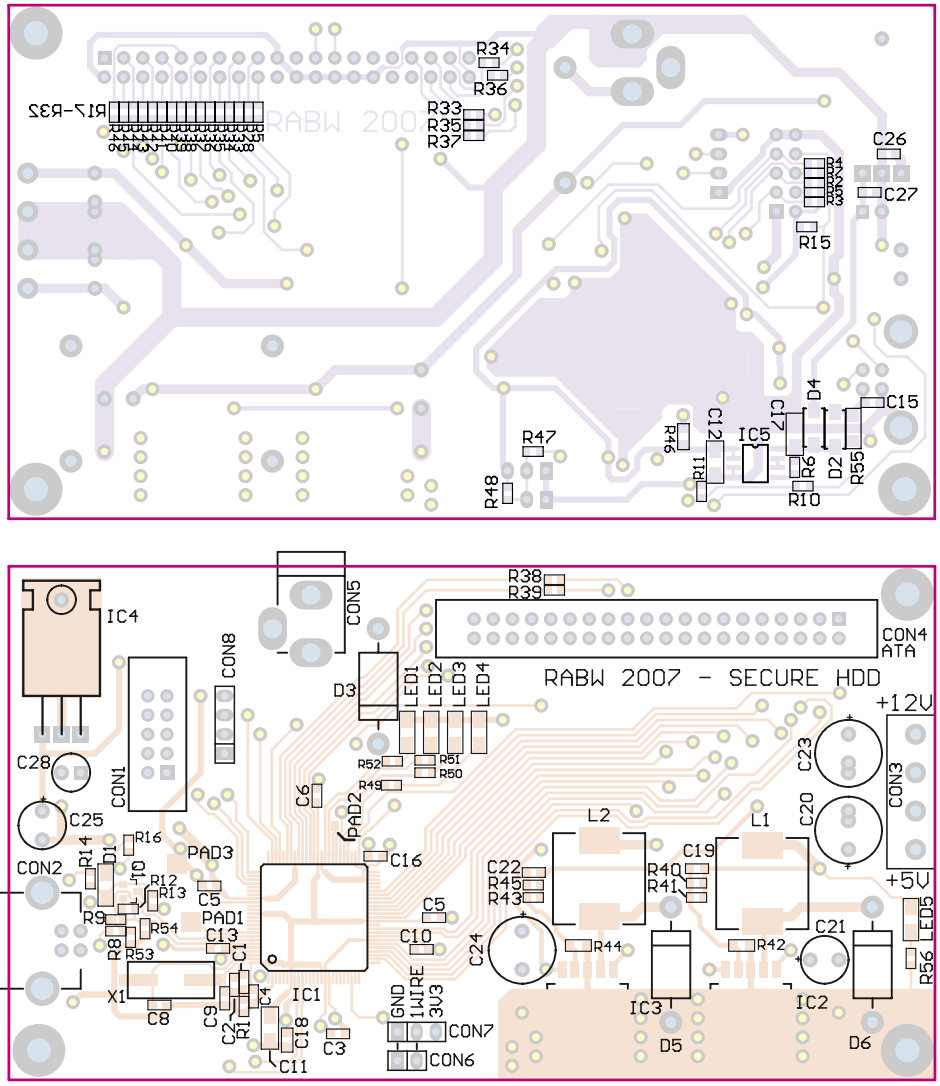
Mozaika ścieżek dysku (**rys. 2**) została opracowana tak, aby możliwe było wykonanie płytki w warunkach amatorskich (np. emulsją światłoczułą), bez metalizacji otworów (każda przelotka może być wykonana za pomocą drutu, a złącza mogą być lutowane wyłącznie od strony *bottom layer*). Tym sposobem został wykonany prototyp.

Warto pamiętać, aby elementy na płytce montować rozpoczynając od bloków zasilania. Po ich wlutowaniu należy bezwzględnie sprawdzić poprawność napięć zasilających dla dysku i dokonać ewentualnej korekty wartości rezystorów R40...R45 ustalających to napięcie. Warto także sprawdzić przed wlutowaniem mikrokontrolera, czy da się włączać i wyłączać zasilanie dysku przy pomocy wyprowadzeń 5 układów IC2 i IC3 (sygnał ATA_PWR na schemacie ideowym) zwierając sygnał ATA_PWR do masy i do napięcia +3,3 V.

Jeśli wszystkie napięcia są poprawne, można przylutować mikrokontroler AT91SAM7XC256. Po kilkukrotnym sprawdzeniu poprawności montażu, można przeprowadzić „próbę dymu" podłączając na kilka sekund układ do zasilania, a następnie przejść do operacji ładowania firmware'u.

Programowanie mikrokontrolera AT91SAM7XC256

Jeśli nie dysponujemy interfejsem JTAG, proces programowania mikrokontrolera można przeprowa-



Rys. 2. Schemat montażowy przejściówki

dzić za pomocą programu SAM–BA. Opis będzie dotyczył SAM–BA v2.6 (lecz wszystkie nowsze wersje także powinny działać) i instalacji dla systemu Windows XP.

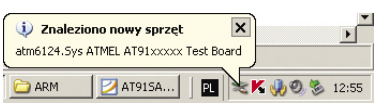
Przed przystąpieniem do wykonywania niżej przedstawionej instrukcji należy mieć zainstalowany pakiet programów AT91–ISP v1.1 lub nowszy (w nim znajduje się program SAM–BA i drivery – wszystko do znalezienia za darmo na stronie *www.atmel.com*). Programowanie najlepiej rozpocząć od profilaktycznego skasowania układu poprzez chwilowe zwarcie nóżki ERASE (wyprowadzenie 92 mikrokontrolera) do szyny zasilania +3,3 V. Najwygodniej można tego dokonać poprzez chwilowe zwarcie padów PAD1 i PAD3 na płytce urządzenia. Kolejne etapy pracy opisano niżej.

1. Wiedząc, że mikrokontroler jest „skasowany" podłączamy wtyczkę USB do portu komputera PC (wystarczy, że urządzenie będzie zasilane tylko z portu USB). Powinien pojawić się windowsowski „dymek" informujący o rozpoznaniu nowego urządzenia (**rys. 3**).
2. Po krótkiej chwili powinien się pojawić Kreator znajdowania nowego sprzętu. Na pytanie „Czy system Windows może połączyć się z witryną..." klikamy jak zwykle odpowiedź Nie, nie tym razem i przycisk Dalej.
3. W następnym okienku kreatora zaznaczamy opcję Zainstaluj oprogramowanie automatycznie (zalecane). Klikamy Dalej.
4. Po niecałej sekundzie powinno ukazać się okienko informujące o tym, że oprogramowanie nie przeszło testów zgodności. Oczywiście bez obaw klikamy mimo to Kontynuuj.
5. W okienku „Kończenie pracy Kreatora..." klikamy Zakończ.
6. Teraz powinien pojawić się kolejny „dymek" z napisem informującym, że zainstalowany sprzęt jest gotowy do użycia.
7. Uruchamiamy program SAM–BA v2.6.
8. Powinno pojawić się okienko wyboru mikrokontrolera. Z listy wyboru *Select your board* wybieramy *AT91SAM7XC256–EK*, okienko powinno wyglądać jak na **rys. 4** (połączenie USB). Klikamy *Connect*.
9. Teraz powinniśmy zobaczyć właściwe okienko SAM–BA. Klikamy zakładkę *Flash* (w środku okienka), a następnie ikonkę otwarcia pliku na lewo od przycisku *Send File*, znajdujemy plik z wersją oprogramowania, które wybraliśmy i klikamy *Send File*.
10. Gdy program SAM–BA zapyta, czy chcemy ustawić *lockbity* na zapisanych obszarach pamięci, postępujemy wg uznania.
11. Po zakończonym procesie programowania, w polu *Scripts* z rozwijanej listy wybieramy *Boot from Flash (GPNVM2)* i klikamy *Execute*.
12. Odłączamy napięcie zasilania układu (przez co zerujemy układ).

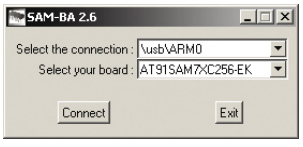
Gotowe! Po włączeniu zasilania przejściówka wystartuje już z nowo wgranym programem. Ponowne programowanie (np. inną wersją) można najłatwiej wykonać albo za pomocą interfejsu JTAG, albo kasując wcześniej wszystkie pamięci nieulotne mikrokontrolera (nóżką ERASE) i powtarzając punkty powyższej instrukcji.

Obsługa dysku zewnętrznego

Jeśli sprzęt jest sprawdzony i działający, a mikrokontroler zaprogramowany, należy podłączyć (w kolejności): dysk twardy z rów-



Rys. 3. „Dymek" informujący o wykryciu mikrokontrolera SAM7 z bootloaderem SAM–BA



Rys. 4. Okno wyboru mikrokontrolera przy starcie programu SAM–BA

noległym interfejsem ATA do złącza CON4, zasilanie dysku (najlepiej za pomocą obciętego kabla od starego zasilacza komputerowego) do złącza CON3 i zasilanie całego urządzenia do złącza CON5. Kolejne etapy zostały opisane niżej.

1. W tym momencie napęd powinien być zasilany, a po chwili powinna włączyć się dioda LED1.
2. Rozpoznajemy wyprowadzenia układu lub układów Dallas, którymi dysponujemy i przykładamy je w ustalonej przez siebie kolejności do złącza CON6. Jeśli zdecydowaliśmy się na wersję V2 oprogramowania, po przyłożeniu pierwszego „Dallas'a" na około 1 sekundę włączy się dioda LED2, a kiedy zgaśnie będzie można przyłożyć drugi układ Dallas (może to być także drugi raz ten sam układ). W przypadku wersji V1 przykładamy tylko 1 układ Dallas.
3. W tym momencie jest już ustawiony klucz do szyfrowania i deszyfrowania zawartości dysku (świeci dioda LED2). Jeśli wcześniej nie włożyliśmy wtyczki USB, można to zrobić także teraz (jeśli nie podamy klucza, nie zostanie nawiązana transmisja USB).
4. Gdy zechcemy zakończyć pracę z dyskiem, po prostu wyjmujemy wtyczkę USB z gniazdka, a wtedy zasilanie napędu zostanie odłączone do następnego włożenia wtyczki USB. Po wyjęciu wtyczki USB klucz jest kasowany, więc będzie należało podać go jeszcze raz przed ponownym włączeniem dysku do komputera PC.

Urządzenie nie sprawdza poprawności klucza podanego układami Dallasa, klucz jest jedynie wpisywany do odpowiednich miejsc w pamięci RAM mikrokontrolera. Dlatego, jeśli sformatujemy dysk twardy używając jednego klucza, a później „otworzymy" dysk innym kluczem, dane na dysku nie będą miały sensu. Nic się jednak nie powinno stać z danymi, jeśli nie sformatujemy dysku z niepoprawnym kluczem (system Windows może zapytać czy chcemy formatować dysk, jeśli nie wykryje na nim poprawnych danych).

Formatowanie dysków można przeprowadzić nie tylko z okna *Mój komputer* lecz także przy pomocy programu Disk Manager (standardowo w *C:\WINDOWS\system32\diskmgmt.msc*). Szczególnie polecam wykonywać formatowanie z Disk Managera, gdy chcemy sformatować duży dysk. W przypadku tego projektu włączenie opcji *szybkie formatowanie* pod względem funkcjonalności nie różni się od standardowego formatowania, lecz trwa krócej.

Polecam używanie dysków twardych bez tzw. *bad sectors* czyli sektorów, z których dane nie mogą zostać poprawnie odczytane. Zaimplementowane zostały jedynie podstawowe funkcje detekcji błędów. Jeśli „zły sektor" wystąpi w obszarze krytycznym dla systemu plików, formatowanie nie zakończy się pomyślnie.

W związku z powolnymi transferami danych rzędu 0,5 MB/s zalecana jest cierpliwość przy starcie sformatowanego dysku, podczas operacji kopiowania danych oraz formatowania dysku, szczególnie w przypadku dysków o pojemnościach rzędu setek gigabajtów.

Metody ataku

Na stronie [8] można uzyskać informacje na temat ostatnio złamanych szyfrów. W momencie pisania tego artykułu skuteczne i uniwersalne algorytmy łamiące AES jeszcze nie istniały. Jedyny dotychczas udany atak na AES polegał na badaniu czasu dostępu do pamięci cache procesora wykonującego szyfrowanie, więc był to tzw. atak typu ***side channel*** (nie bazujący na kryptoanalizie, lecz na niedoskonałościach fizycznej implementacji algorytmu szyfrowania). Atak ten można przeprowadzić na urządzeniach, na których istnieje możliwość uruchamiania programów z zewnątrz, więc nie dotyczy on opisywanej tutaj przejściówki.

Zakładamy, że „intruz" także czyta EP, więc wie, iż przedstawione tutaj urządzenie szyfruje dane algorytmem AES. Dopuszczamy także reverse–engineering i dowolne metody badania kodu programu – są to metody zupełnie bezskuteczne, ponieważ urządzenie w swoim mikrokontrolerze nie ma zakodowanego nic „tajnego" (algorytm AES **używa klucza**, ale w żadnym wypadku **nie porównuje klucza**). Dalej omówione zostaną tylko ataki typu ***brute force*** polegające na wyczerpującym poszukiwaniu klucza.

„Intruz", chcąc odszyfrować dysk może odczytać fragment któregoś sektora, a następnie próbować odszyfrować ten fragment różnymi kombinacjami bitów klucza. Będzie miał 2^{128} różnych kluczy do sprawdzenia. Niestety, ponieważ „straciliśmy" nieco bitów przez 56–bitowe numery seryjne (48–bitowy numeru + 8–bitowy *Family Code*, CRC można obliczyć) otrzymujemy 2^{112} możliwości (dla wersji bezpieczniejszej). Można szacować, że średni czas złamania szyfru w dysku wyniesie około $8 \cdot 10^{19}$ lat przy założeniu, że jedna próba będzie trwała jedną mikrosekundę (10^{-6} s). Dla wersji mniej bezpiecznej (2^{56} kluczy), czas ten wyniesie jedynie około 1100 lat. Mimo to, zabezpieczenia z kluczami 56–bitowymi w obecnych czasach uznaje się za podatne na ataki *brute force* – taki atak staje się opłacalny przy zastosowaniu odpowiednio szybkiego komputera. W ten sposób „poległ" algorytm DES (*Data Encryption Standard*, poprzednik AES) z 56–bitowym kluczem.

Przedstawione powyżej czasy złamania zabezpieczeń są obliczone przy założeniu, że każda kombinacja bitów klucza jest jednakowo prawdopodobna (rozkład jednostajny). Należy pamiętać, że do dysku można się włamać nawet za pierwszą próbą – prawdopodobieństwo takiej sytuacji jest bardzo małe, lecz nie zerowe (równe 1/*liczba kluczy*).

Dobrym pomysłem może się okazać zbudowanie urządzenia przechowującego klucz i symulującego obecność układu firmy Dallas. Urządzenie można wykonać np. na małym, tanim mikrokontrolerze implementując funkcje 1–Wire slave. Dzięki temu można będzie wpisywać dowolny klucz i ewentualnie go skopiować dla innych uprawnionych użytkowników.

Alternatywy softwarowe

W sytuacjach, gdy zaprezentowana tutaj przejściówka z szyfrowaniem AES będzie miała zbyt

małą szybkość działania, można rozważyć zastosowanie programów realizujących tzw. szyfrowanie danych w locie, czyli OTFE (*On The Fly Encryption*). Intensywnie rozwijanym programem szyfrującym jest dostępny za darmo TrueCrypt [7]. Realizuje on operacje szyfrowania algorytmami AES, Serpent i Twofish oraz ich kombinacjami.

Oczywiście stosowanie programów OTFE ma swoje zalety i wady. W specyficznych sytuacjach, sprzętowe szyfrowanie danych przy pomocy opisanej tutaj przejściówki może być bardzo przydatne. Np. stosując przejściówkę uniemożliwiamy „podsłuchanie" klucza, ponieważ nie wprowadzamy hasła z klawiatury – unikamy niektórych ataków typu *side channel*. Inną zaletą rozwiązania hardware'owego jest absolutna przezroczystość szyfrowania dla systemu operacyjnego. Oznacza to, że w systemie nie musimy mieć zainstalowanych ani uruchomionych żadnych programów realizujących szyfrowanie.

Ważne uwagi

1. **Projekt NIE ZOSTAŁ opracowany, aby komukolwiek umożliwić bezkarne przechowywanie danych niezgodnych z prawem**. Przejściówka ma służyć jedynie eksperymentalnej ochronie prywatnych danych przed niepowołanymi osobami. Z racji niezbyt dużej szybkości działania, przejściówka w założeniach została zaprojektowana do wygodnej (bez podawania haseł) archiwizacji dokumentów, projektów, korespondencji e–mail, czy notatek.
2. Autor dołożył wszelkich starań, aby urządzenie opisane w artykule działało poprawnie, lecz nie ponosi odpowiedzialności za jakiekolwiek straty danych czy uszkodzenia sprzętu wynikłe pośrednio lub bezpośrednio z faktu posiadania, uruchamiania i używania opisanego tutaj urządzenia.
3. Źródła programu sterującego (język C, kompilator GCC), co prawda nie są publikowane oficjalnie, lecz mogą zostać udostępnione wszystkim zainteresowanym. Jeśli Czytelnik chce udoskonalić, zmodyfikować przejściówkę lub po prostu lepiej poznać jej działanie, proszę o kontakt.

Robert Brzoza–Woch
rabw@poczta.fm

Literatura

[1] http://www.partow.net/projects/galois/
[2] http://www.csrc.nist.gov/publications/fips/fips197/fips–197.pdf
[3] http://en.wikipedia.org/wiki/Advanced_Encryption_Standard
[4] http://www.atmel.com/dyn/resources/prod_documents/doc6209.pdf lub prościej: www.atmel.com i w okienku wyszukiwania „AT91SAM7XC256"
[5] R. Brzoza–Woch „Czytnik kart SD", EP 4/2007
[6] AT Attachment with Packet Interface – 6 (ATA/ATAPI–6) Rev. 3B, www.t13.org
[7] http://www.truecrypt.org/
[8] http://www.cryptosystem.net/aes/