Armeluis chiptweaker

we krijgen je wel (goedkoop) aan de praat...

By Luka Matic (Croatia)

Hardware-beveiligingsmodules (hardware security modules, HSM's) zijn elektronische apparaten die gebruik maken van verschillende hardware-beveiligingen tegen het ongeoorloofd lezen van de geheime gegevens die ze bevatten. Zij voeren gewoonlijk handelingen uit zoals versleuteling of digitale ondertekening, maar kunnen ook andere taken uitvoeren. Bankkaarten of toegangskaarten zijn typische voorbeelden van HSM's. Deze moeten pincodes en privésleutels zo lang mogelijk geheim houden voor het geval de kaart wordt gestolen of verloren gaat. Bovendien hebben veel MCU's voor algemeen gebruik ook geheugenbeschermingsfuncties om het illegaal kopiëren van de eigen firmware te voorkomen. De hier gepresenteerde armeluis chiptweaker is een tool om de beginselen van niet-invasieve aanvallen op HSM's te leren kennen. Hiermee kun je proberen een microcontroller met geheugenbeveiliging te ontgrendelen.

Zoals uitgelegd in een vorig artikel [1], kunnen onze arme spionnen Alice en Bob er niet op hopen hun eigen HSM's te bouwen, omdat daarvoor dure gespecialiseerde apparatuur en kennis vereist zijn waarover ze waarschijnlijk niet beschikken. Voor hun geheime operaties gebruiken ze hun eigen hardware, gebouwd met goedkope en gemakkelijk verkrijgbare componenten waarop ze kunnen vertrouwen. Dit werkt goed voor hen, zolang de juiste Operations Security-procedures (OpSec) worden gevolgd. Aan de andere kant kunnen ze ook niet om het gebruik van HSM's zoals bankkaarten heen, en ze kunnen op een HSM stuiten waarvan ze de geheimen te weten willen komen. De armeluis chiptweaker (Poor Man's ChipTweaker, PMCT) zal daarom heel goed van pas komen, omdat het hen in staat stelt de principes van niet-invasieve aanvallen op HSM's te leren.

Hardware-aanvallen

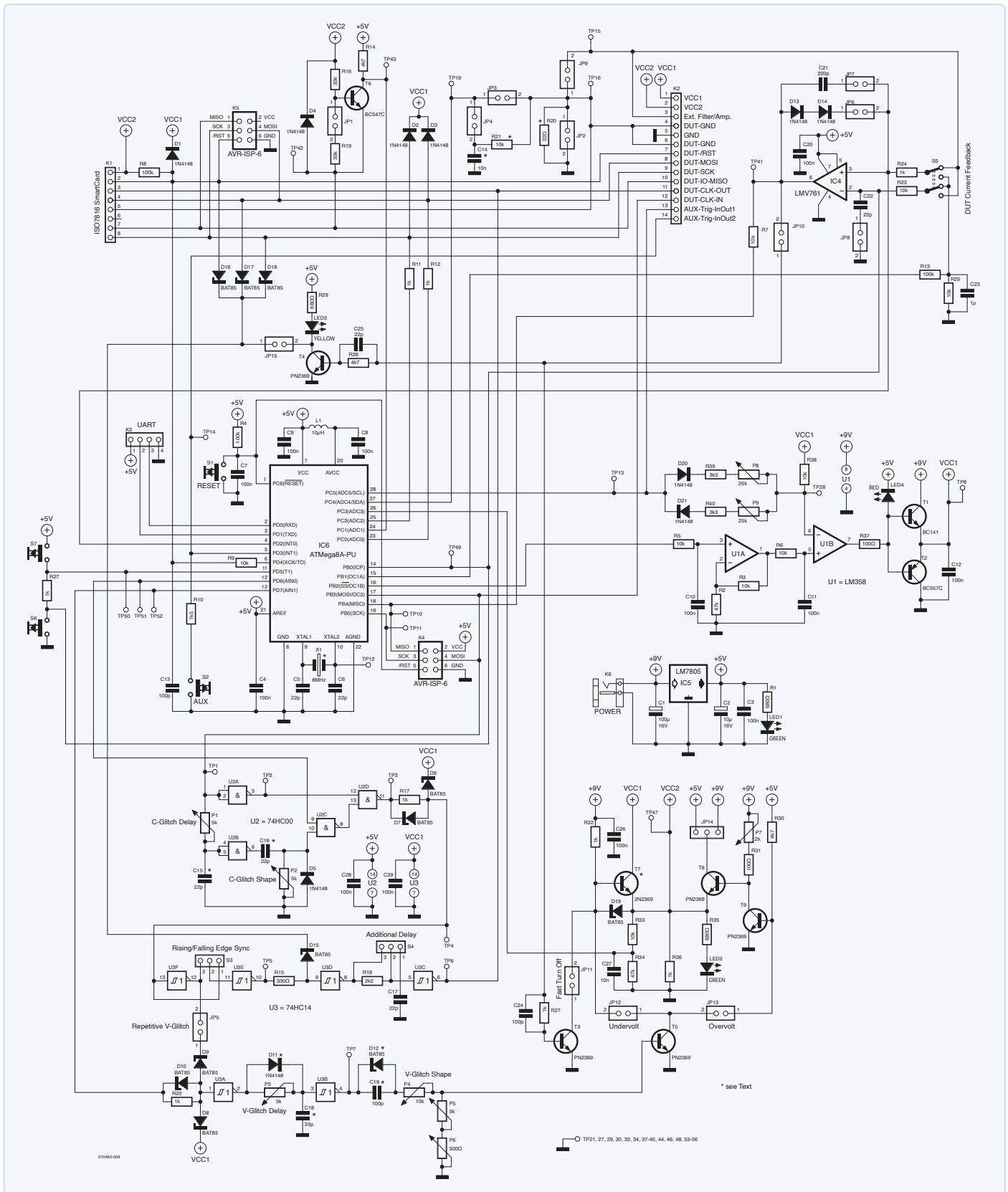
Bij aanvallen op HSM's kunnen we drie typen onderscheiden:

1. **Niet-invasief:** de HSM wordt niet geopend of uit elkaar gehaald. De aanval wordt uitgevoerd met behulp van verschillende elektrische signalen op de gewone communicatie- en voedingsaansluitingen van de HSM. Hiervoor is de PMCT ontworpen. Dit is een goedkope manier.
2. **Semi-invasief:** de afdekking van het halfgeleiderkristal van de HSM wordt verwijderd. De elektrische contacten op de silicium-*die* worden niet blootgelegd. De aanval wordt uitgevoerd door bepaalde delen van de HSM-microschakelingen aan lichtpulsen bloot te stellen. Deze manier is al duurder.
3. **Invasief:** het halfgeleiderkristal ligt volledig bloot, zodat microprobes kunnen worden aangesloten op de contacten op de *die*. De aanval wordt uitgevoerd door elektrische signalen in de HSM-microschakelingen te injecteren. Dit is een kostbare aanval waarvoor zeer gespecialiseerde apparatuur nodig is.

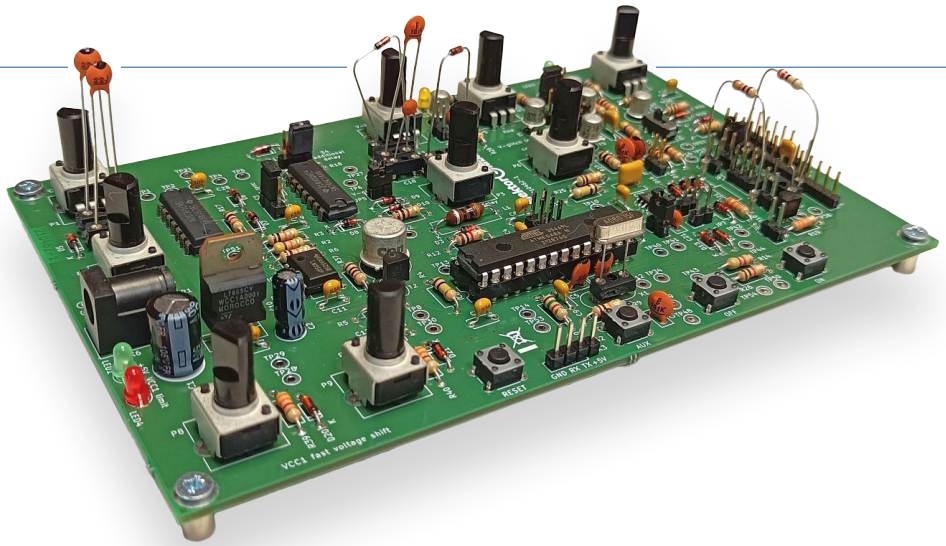
Er zijn natuurlijk krachtige digitale *ChipWhisperers* verkrijgbaar [2]. Maar wat ik wilde doen was iets maken op basis van oudere technologie, vergelijkbaar met wat dr. Skorobogatov heeft gedaan [3], waarbij alle analoge en digitale signalen volledig toegankelijk zijn en wat wellicht een betere manier is om de principes van niet-invasieve aanvallen te begrijpen.

Specificaties

Niet-invasieve aanvallen zijn gebaseerd op verschillende foutinjectiemethoden, zoals het verzenden van slecht geformatteerde gegevens of het verzenden van signalen met een onjuiste amplitude en frequentie naar een van de pinnen van de HSM (inclusief de voedingspinnen). Deze kunnen de DUT (*Device under Test*, of beter *Device under Attack*) vele ongecontroleerde acties laten uitvoeren en hopelijk zijn geheimen doen prijsgeven.
Niet-invasieve aanvallen vereisen veel gegevensverwerking, tweaken en jagen in het donker voor elke specifieke DUT, maar als het mogelijk is om hem niet-invasief te kraken, kan de goedkope en eenvoudige PMCT je daarbij helpen. Hij heeft de volgende mogelijkheden:



Figuur 1. Het schema van de Poor Man's ChipTweaker. Let op het gebruik van snelle transistoren van het type 2369. Om redenen van warmteafvoer moet T7 een TO-18 type zijn (metalen behuizing, 2N2369). De plastic PN2396-types die voor de andere worden gebruikt zijn wellicht goedkoper en/of gemakkelijker te vinden. Andere met een '' aangegeven onderdelen moeten in een busstrip worden gemonteerd om gemakkelijk te kunnen experimenteren met verschillende waarden.*



Figuur 2. De Poor Man's ChipTweaker is opgebouwd op een comfortabele print. De in busstrips gemonteerde onderdelen zijn in het schema aangegeven met ''.*

1. **Klokglitch-generator:** het injecteren van te snelle signalen op de klokingang (CLK) van de HSM kan ervoor zorgen dat de CPU een machine-instructie overslaat of verkeerd uitvoert, en dus bijvoorbeeld een PIN-controle overslaat.
2. **Spanningsglitch-generator:** het wijzigen van de voedingsspanning van de HSM tot buiten het nominale bereik kan ook een ongecontroleerde actie veroorzaken. De PMCT kan langzame, middellange en snelle spanningsglitches leveren, zowel onderspanning als overspanning.
3. **Variabele 1,0...6,0 V DUT-voeding:** gestuurd door de microcontroller van de PMCT.
4. **Snelle DUT-turn-off comparator:** boven of onder een bepaalde voedingsstroomdrempel (voor bijvoorbeeld een EEPROM-schrijfsabotage-aanval), armed-disarmed of direct getriggerd door de hoofd-MCU.
5. **Bit-banged SPI en bidirectionele een-pin UART-interface** voor communicatie met de DUT (inclusief alle typen SmartCards), met bidirectionele level shifters in het bereik van 1,5 V tot 6,0 V.

Naast bovengenoemde aanvallen kan de PMCT in combinatie met een digitale 'scoop (zoals een LabNation SmartScope), een 100...200 MHz analoge geheugenscoop (zoals de Tektronix 466) en een PC met software als MATLAB voor offline gegevensanalyse allerlei soorten timing- en voedingsanalyses uitvoeren.

In tegenstelling tot de volledig digitale ChipWhisperers zijn alle analoge en digitale signalen toegankelijk voor studie en analyse, en kan de glitch-timing nog nauwkeuriger worden afgesteld dankzij de continue analoge afstelling met potentiometers zonder kwantiseringsstapjes. Daarom kan een trage ATmega8 het hele aanvalsproces coördineren, en is er geen FPGA nodig. Glitches met een duur van minder dan 10 ns kunnen worden gegenereerd; deze veroorzaken fouten in CPU's die zijn ontworpen voor frequenties tot 50 MHz en hoger. Veel SmartCards en MCU's werken slechts tot maximaal 20 MHz.

Beschrijving van de hardware

In het midden van het schema (**figuur 1**) zien we de microcontroller-eenheid (MCU, IC6, ATmega8) die de aanvalsprocessen coördineert. De schakeling rond IC1 is een variabele spanningsbron voor de DUT. Zijn uitgangsspanning VCC1 wordt geregeld via PWM door timer-uitgang OC1B van IC6. Hierdoor kunnen langzame spanningsschommelingen worden opgewekt. MCU-pen PC5 kan een van drie toestanden hebben – L, H en high-Z – waardoor hij de versterking van IC1B snel kan veranderen en dus drie verschillende waarden van VCC1 kan produceren, ingesteld door P8 en P9. Dit is de manier om een doorsnee spanningsglitch (orde van grootte van µs) te genereren. De rode LED4 gaat branden als waarschuwing en beperkt VCC1 tot ongeveer 6 V, wat niet te hoog is voor een 5V-apparaat.

Glitches genereren

De DUT wordt normaal gesproken gevoed door VCC2 via transistor T7 (een 2N2369 in een TO-18 behuizing). T5 kan diens basis snel laag trekken (plaats jumper JP12), wat resulteert in een snelle onderspanning-glitch (in de orde van grootte van 10 ns). Als JP13 wordt geplaatst, zal T8 VCC2 snel hoog trekken, waardoor een snelle overspanning-glitch ontstaat. Snelle spanningsgolven worden gegenereerd door C19, P4, P5 en P6. De vertraging van de spanningsglitch ten opzichte van de CLK-puls wordt ingesteld met P3. Als MCU-uitgang PD7 hoog wordt gemaakt, wordt een enkele spanningsglitch afgegeven. Als deze hoog blijft, en JP5 wordt kortgesloten, worden bij elke CLK-puls spanningsglitches afgevuurd.

IC2 wordt gebruikt om de DUT op een klokglitch te trakteren. De PWM-uitgang OC2 van de MCU genereert de CLK-puls voor de DUT. Als poort PD6 hoog is, geeft IC2C bij elke CLK-puls een korte glitch door aan IC2D. De polariteit van de glitch kan worden gekozen met een jumper of schakelaar op S3.

IC4 is een snelle comparator met positieve terugkoppeling (versterkt door D13, D14 en C21) die de DUT in een paar ns kan uitschakelen. Dit kan ook worden bereikt met een commando van de MCU (PD5 of PD2 om in te schakelen, PB0 om de DUT uit te schakelen) of door op de knoppen S6 en S7 te drukken.

Aanval via het stroomverbruik

IC4 kan ook worden geconfigureerd om de DUT uit te schakelen als het stroomverbruik van de DUT (gemeten overt weerstand R20) hoger of lager (geselecteerd met S5) is dan een drempelwaarde (bij C23, gemeten via PWM op de OC1A pen). Dit wordt typisch gebruikt om te voorkomen dat de DUT naar zijn interne EEPROM schrijft (wat extra stroom verbruikt). De uitgang van IC4 activeert T3, die T7 uitschakelt. Hij activeert ook transistor T4 (auxiliary turn-off) om alle SmartCard-lijnen laag te trekken en mogelijke fantoom-terugvoeding te voorkomen.

Analoge ingang ADC4 bewaakt de stroom van de DUT. Aangezien dit een langzame ADC is, kan het signaal worden gefilterd door R21/C14, of het kan eventueel nog worden verwerkt door een sneller extern filter/versterker (op connector K2). Voor een timingaanval of vermogensanalyse wordt dit signaal meestal opgenomen door een oscilloscoop voor offline verwerking en analyse met een programma als MATLAB.

Aangezien de MCU op 5 V werkt terwijl de DUT op de (meestal) lagere spanning VCC2 werkt, is de bidirectionele level shifter (rond T6) nodig voor de I/O-pin 7 van de DUT op connector K1 (een bidirectionele UART-poort, zoals de meeste SmartCards tegenwoordig gebruiken). Pinnen 4 en 8 worden gebruikt

voor SmartCards met SPI-interface (bijvoorbeeld de FunCard). Omdat ze unidirectioneel zijn, zijn hun level shifters eenvoudiger (diodes D2 en D3).

De print

Om de bouw van je eigen Poor Man's ChipTweaker te vergemakkelijken is er een print ter grootte van een eurokaart voor ontworpen (**figuur 2**). (Zie [4] voor het KiCad-project en de Gerber-bestanden.) Alle onderdelen zijn van het through-hole type, behalve IC4, dat een SOIC-8 behuizing heeft. Het volbouwem van de print zou geen problemen mogen opleveren, maar er zijn enkele zaken waar je op moet letten:

- In plaats van de 3-polige jumpers 'Sxx' zijn SPDT schuif- of wipschakelaars wellicht handiger.
- Misschien gebruik je liever liever instelpotmeters dan gewone potmeters. Trimmers zijn gemakkelijker verkrijgbaar in meer waarden.
- T7 moet een 2N2369 zijn in een metalen TO-18 behuizing in verband met de warmteafvoer. De PN2396-transistoren in TO-92 behuizing kunnen worden vervangen door 2N2369-types, afhankelijk van wat je te pakken kunt krijgen.
- Onderdelen met een sterretje (*) (zoals D11 en C18, enzovoort, met uitzondering van T7) worden verondersteld in een busstrip te worden gemonteerd om gemakkelijk te kunnen experimenteren met verschillende waarden en types.

Het board heeft veel testpunten, altijd met een massaverbinding in de buurt. Probeer de nummering ervan niet te begrijpen; daar ontbreekt elke logica.

Aanvallen!

De DUT voor onze demo-toepassingen is een standaard FunCard (met een Microchip AVR AT90S8515 MCU plus een AT24C EEPROM). Zonder serieuze beveiligingen is een aanval relatief eenvoudig. Laad de FunCard met de firmware van [4] via ISP-connector K5. Gebruik vervolgens een van de twee ATmega8 firmware-varianten (voor het aanvallen van een FunCard, ook bij [4]) om de MCU IC6 van de ChipTweaker te programmeren via ISP-connector K4 (en niet K5). Je kunt nu wat basisprocedures uitproberen.

Sluit een seriële terminal aan op de UART-poort K3 en configureer deze voor 9600-8-N-1 (of 9600n81 zo je wilt). Nadat je op de reset-knop S1 hebt gedrukt, zou je een scherm moeten krijgen zoals in **figuur 3**. De opties 0 en 1 zijn voor het versturen van eenvoudige commando's naar de MCU en de FunCard. Alle commando's zijn 4 bytes lang, waarbij de laatste byte altijd gelijk is aan 0x0d (CR). Zie **tabel 1** en **tabel 2**.

Demo: klok-glitch aanval

Dit is een demonstratie van een eenvoudige klok-glitch aanval. De FunCard voert verschillende 16-bit bewerkingen uit op twee ingangs-operanden die zich in de interne EEPROM van de MCU bevinden op de adressen 0x00, 0x01, 0x02 en 0x03 (in big-endian vorm). Het glitchen van een klokpuls (het sturen van een puls die te snel is voor de DUT)

```
> 0 : Send a single command to Smartcard
> 1 : Send a single command to MCU
> 2 : Find minimum Vcc2
> 3 : Adjust minimum Vt
> 5 : Funcard no glitch demo
> 6 : Funcard clock glitch demo
> 7 : Funcard volt glitch demo
> 8 : Volt glitch loop test
> 9 : Volt & Clock glitch loop test
> A : Fast Vcc2 test
> B : Brute-force PIN demo, Funcard MCU internal counter
> b : Brute-force PIN demo, Funcard AT24C external counter
> R : Reset MCU & Smartcard

Vcc2 RAW set to:0xDD    Vcc2 feedback: 5,092V
Vt RAW set to:0xFF    Vt scaled set to: 0,452V    Vi feedback: 0,166V
Smartcard fclk division factor set to RAW:0x00

> Select Option (0-X): 5

No glitch demo.

Answer To Reset: 0x45 0x67 0x78 0x9A 0xBC 0xDE 0xF0 0x12 0x34 0x56 0x78 0xDE 0xAD 0x01 0x23 0x00 0x00 0x00 0x00 0x00
```

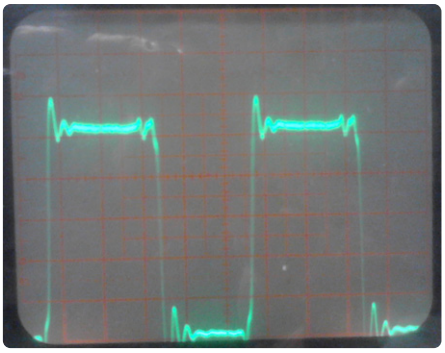
Figuur 3. Selecteer de aanval die je wilt uitvoeren uit een menu.

Commando	Beschrijving
V2x	'x' is een 8bit-getal dat de PWM-verhouding voor OC1B bepaalt om de VCC2-spanning in te stellen.
Vtx	'x' is een 8bit-getal dat de PWM-verhouding voor OC1A bepaalt om de Vt-spanning in te stellen, een drempel voor C23 om IC4 te activeren.
Fxy	'x' is een ruwe frequentie-deelfactor, voor het genereren van een CLK-puls voor de DUT op OC2. Indien ingesteld op 0, loopt de OC2 op de helft van de MCU-kristalfrequentie; 'y' wordt genegeerd.
GLx	De laagste 3 bits van 'x' stellen de MCU-pinnen PD7, PD6 & PC5 in; voor testdoeleinden.
onx	Schakel de DUT in; 'x' wordt genegeerd.
ofx	Schakel de DUT uit; 'x' wordt genegeerd.
Sxy	Sla de huidige instellingen op in de EEPROM van de MCU; 'x' en 'y' worden genegeerd.

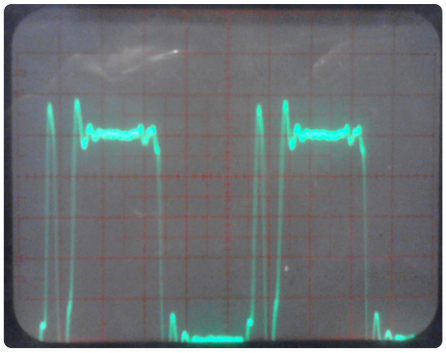
Tabel 1. Commando's voor de MCU. Elk commando wordt met 0x0d (CR) afgesloten.

Commando	Beschrijving
Rxy	Lees een byte van adres 'x' van de interne EEPROM van de AT90S; 'y' wordt genegeerd.
Wxy	Schrijf byte 'y' naar adres 'x' van de interne EEPROM van de AT90S.
Pxy	Test een PIN-code 'xy' (in BCD-formaat). De PIN wordt opgeslagen in de interne EEPROM van de AT90S.
rxy	Lees een byte van adres 'x' van de AT24C-EEPROM; 'y' wordt genegeerd.
wxy	Schrijf byte 'y' naar adres 'x' van de AT24C-EEPROM.
pxy	Test een PIN-code 'xy' (in BCD-formaat). De PIN is opgeslagen in AT24C-EEPROM.

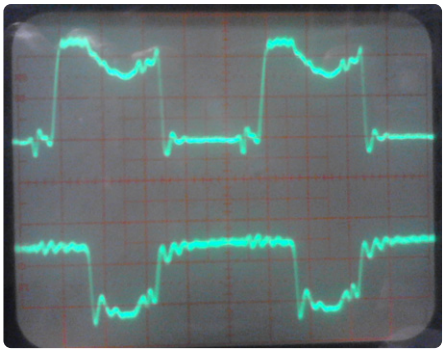
Tabel 2. Commando's voor de SmartCard. Elk commando wordt met 0x0d (CR) afgesloten.



Figuur 4. Een schoon kloksignaal zonder glitches. De oscilloscoop was ingesteld op 50 ns/div op de horizontale as en 1 V/div op de verticale as.



Figuur 5. Een kloksignaal met glitches (50 ns/div horizontaal, 1 V/div verticaal). Regel potentiometers P1 en P2 af om een glitch te maken zoals hier getoond.



Figuur 7. De spanning-glitch komt 30 tot 40 ns na de opgaande flank van de klokpuls. Het bovenste spoor is het kloksignaal, het onderste is de voedingsspanning VCC2 (50 ns/div horizontaal; 1 V/div verticaal).

in verschillende stadia van de berekeningen levert verschillende fouten op. Misschien is het resultaat van de wiskundige bewerking verkeerd, of wordt er gewoon een machine-instructie overgeslagen.
Zie **figuur 4**, **figuur 5** en **figuur 6** voor de signalen en de resultaten. Een effectieve glitch voor de FunCard is een actief lage puls met een duur tussen 10 ns en 20 ns die ongeveer 10...20 ns na de opgaande flank van de CPU-klok wordt afgevuurd (zie figuur 5).

Demo: spanning-glitch aanval

Dit is een demo van een eenvoudige aanval met een spanning-glitch. De FunCard voert dezelfde bewerkingen uit als in het vorige voorbeeld. Een effectieve onderspanning-glitch heeft een duur van minstens 50 ns en levert goede resultaten op met de timing van figuur 7. Hier begon de glitch 30...40 ns na de opgaande flank van de klokpuls. De voedingsspanning voor de FunCard was ingesteld op 2,24 V. De spanningsdip was ongeveer 1,5 V diep.
Stel de vorm van de spanningstoename in met P4, P5 en P6. Gebruik P3, S3 en S4 om de glitchvertraging en -synchronisatie in te stellen. De resultaten worden op dezelfde manier weergegeven als in het vorige voorbeeld (**figuur 8**).

De geheugenbeveiliging kraken om de firmware te extraheren

Elke beschermde MCU of SmartCard vereist een andere aanvalsprocedure. Het gaat gepaard met veel vallen-en-opstaan en jagen in het donker om mogelijke zwakke plekken te vinden. De procedure die werkt op veel Microchip AVR-microcontrollers is een langzame onderspanning-glitch.
Dat gaat als volgt. Wanneer de FunCard MCU (een AT90S8515) beschermd is, worden beide memory lock-bits op 0 geprogrammeerd. Het flash-geheugen kan dan niet worden gelezen, alleen gewist. In seriële programmeermodus (de reset-ingang naar 0 V getrokken) zal het *Chip Erase*-commando eerst het flash-geheugen wissen (alle flash-bytes resetten naar 0xFF) en dan de lock-bits wissen (uitschakelen door ze op 1 te zetten).
De ontwerpfout die in veel AVR-MCU's (niet alle) aanwezig is, stelt ons in staat de beveiliging op de volgende manier te omzeilen. Als de voedingsspanning wordt verlaagd tot onder het nominale minimum (2,7 V), en wel tot 1,6...1,7 V, is het niet meer mogelijk om het flash-geheugen te wissen, maar het wissen van de lock-bits lukt nog wel. De aanval wordt gestart bij 1,1 V en de spanning wordt geleidelijk opgevoerd. De lock-bits werden met succes verwijderd bij 1,62 V zonder het flash-geheugen te wissen!
Wanneer je probeert de firmware of de apparaathandtekening te lezen van een beschermde AVR-microcontroller, zal het antwoord "0x00, 0x01, 0x02, 0x03" zijn en dan weet je dat het geheugen beschermd is (**figuur 9**). Zodra je de juiste handtekening krijgt ("0x1E, 0x93, 0x01" voor de AT90S8515) zijn de memory lock-bits gewist en kan het programmageheugen worden gelezen met elke ISP-programmer.

```
> 0 : Send a single command to Smartcard
> 1 : Send a single command to MCU
> 2 : Find minimum Vcc2
> 3 : Adjust minimum Vt
> 5 : Funcard no glitch demo
> 6 : Funcard clock glitch demo
> 7 : Funcard volt glitch demo
> 8 : Volt glitch loop test
> 9 : Volt & Clock glitch loop test
> A : Fast Vcc2 test
> B : Brute-force PIN demo, Funcard MCU internal counter
> b : Brute-force PIN demo, Funcard AT24C external counter
> R : Reset MCU & Smartcard

Vcc2 RAW set to:0xDD    Vcc2 feedback: 5,092V
Vt RAW set to:0xFF    Vt scaled set to: 0,452V    Vi feedback: 0,166V
Smartcard fclk division factor set to RAW:0x00

> Select Option (0-X): 6

Clock glitch demo.

Answer To Reset: 0x45 0x67 0x78 0x9A 0xBC 0xDE 0xF0 0x12 0x34 0x56 0x78 0xDE 0xAD 0x01 0x23 0x00 0x00 0x00 0x00 0x00
0xF62F 0x01 0x25 0x25  OK!
0xF34B 0x02 0x3E 0x3E  OK!
0xF62F 0x03 0x25 0x25  OK!
0xF62F 0x04 0x25 0x25  OK!
0xF5E4 0x05 0xD9 0xD9  OK!
0xF62F 0x06 0x25 0x25  OK!
0xF62F 0x07 0x25 0x25  OK!
0xF576 0x08 0x6B 0x6B  OK!
0x48E8 0x09 0x30 0x30  OK!
0x47E8 0x0A 0x2F 0x2F  OK!
0xF676 0x0B 0x6C 0x6C  OK!
0x47E8 0x0C 0x2F 0x2F  OK!
0xF62F 0x0D 0x25 0x25  OK!
0x0000 0x0E 0x00 0x00  OK!
0x5555 0x0F 0x55 0xAA  Error!
0xF62F 0x10 0x25 0x25  OK!
0xF62F 0x11 0x25 0x25  OK!
0xF62F 0x12 0x25 0x25  OK!
0xF62F 0x13 0x25 0x25  OK!
```

*Figuur 6. Terminalvenster met de resultaten van de klok-glitch aanval. **Kolom 1** is het 16bit-resultaat van de wiskundige bewerking zoals uitgevoerd door de DUT. Het is fout als de glitch succesvol was. **Kolom 2:** glitch-vertraging. Glitches op verschillende tijdstippen geven verschillende resultaten. **Kolom 3:** checksum als de som van de twee bytes van het 16bit-resultaat zoals berekend door de FunCard. **Kolom 4:** checksum berekend door de MCU van de PMCT. Kolom 5: **'Fout!'** als kolommen 3 en 4 niet overeenkomen.*

Voedingsanalyse bij een bankkaart

Bankkaarten worden steeds geavanceerder en gebruiken meerdere methoden om kritieke geheugengebieden fysiek te beschermen. Ze proberen ook de voedingsstroom te maskeren, zodat kritieke handelingen niet gemakkelijk kunnen worden achterhaald. Als het tijdstip van bepaalde procedures, zoals het controleren van de ingevoerde PIN-code, precies kan

```
> 0 : Send a single command to Smartcard
> 1 : Send a single command to MCU
> 2 : Find minimum Vcc2
> 3 : Adjust minimum Vt
> 5 : Funcard no glitch demo
> 6 : Funcard clock glitch demo
> 7 : Funcard volt glitch demo
> 8 : Volt glitch loop test
> 9 : Volt & Clock glitch loop test
> A : Fast Vcc2 test
> B : Brute-force PIN demo, Funcard MCU internal counter
> b : Brute-force PIN demo, Funcard AT24C external counter
> R : Reset MCU & Smartcard

Vcc2 RAW set to:0x60    Vcc2 feedback: 2,238V
Vt RAW set to:0xFF    Vt scaled set to: 0,452V    Vi feedback: 0,053V
Smartcard fclk division factor set to RAW:0x00

> Select Option (0-X): 7

Volt glitch demo.

Answer To Reset: 0x45 0x67 0x78 0x9A 0xBC 0xDE 0xF0 0x12 0x34 0x56 0x78 0xDE 0xAD 0x01 0x23 0x00 0x00 0x00 0x00 0x00
0xF62F 0x01 0x25 0x25  OK!
0xF62F 0x02 0x25 0x25  OK!
0xF5E4 0x03 0xD9 0xD9  OK!
0xF62F 0x04 0x25 0x25  OK!
0x3F4B 0x05 0x8A 0x8A  OK!
0xF62F 0x06 0x25 0x25  OK!
0xF62F 0x07 0x25 0x25  OK!
0x482F 0x08 0x77 0x77  OK!
0x492F 0x09 0x78 0x78  OK!
0x482F 0x0A 0x77 0x77  OK!
0xF676 0x0B 0x6C 0x6C  OK!
0x482F 0x0C 0x77 0x77  OK!
0xF62F 0x0D 0x25 0x25  OK!
0x0000 0x0E 0x00 0x00  OK!
0xF642 0x0F 0x38 0x38  OK!
0x0000 0x10 0x00 0x00  OK!
0xF62F 0x11 0x2F 0x25  Error!
0xF62F 0x12 0x25 0x25  OK!
0xF62F 0x13 0x25 0x25  OK!
```

Figuur 8. Terminalvenster met de resultaten van de spanning-glitch aanval. De kolommen zijn op dezelfde wijze opgemaakt als voor de klok-glitch aanval (zie figuur 6).

CLEAR | AutoScroll | Reset Cnt | 13 | Cnt = 1721 | HEX / ASCII | StartLog | StopLog | Req

```
Stage:0x01 Vcc2 RAW:0x43 Init:0x53
    Reading firmware:0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09
Reading signature... 0x00 0x01 0x02 0x03
Y-erase the lockbits, N-read again, F-finish?
    Vcc2 feedback: 1,575V

Stage:0x01 Vcc2 RAW:0x44 Init:0x53
    Reading firmwar03 03 x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09
Reading signature... 0x00 0x01 0x02 0x
Y-erase the lockbits, N-read again, F-finish?
    Vcc2 feedback: 1,598V

Stage:0x01 Vcc2 RAW:0x45 Init:0x53
    Reading firmware:0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09
Reading signature... 0x00 0x01 0x02 0x03
Y-erase the lockbits, N-read again, F-finish?
    Vcc2 feedback: 1,622V

Stage:0x01 Vcc2 RAW:0x46 Init:0x53
    Reading firmware:0x0C 0x1C 0x1A 0x19 0x18 0x17 0x22 0x2C 0x14 0x13
Reading signature... 0x1E 0x93 0x01 0xFF
Y-erase the lockbits, N-read again, F-finish?

Stage:0x01 Vcc2 RAW:0x47 Init:0x53
    Reading firmware:0x0C 0x1C 0x1A 0x19 0x18 0x17 0x22 0x2C 0x14 0x13
Reading signature... 0x1E 0x93 0x01 0xFF
Y-erase the lockbits, N-read again, F-finish?
```

Transmit

Figuur 9. Het kraken van de beveiliging van het flash-geheugen.

Figuur 10. Voedingsstroom (geel spoor) van een bankkaart bij controle van een onjuiste PIN-code.

worden vastgesteld, dan weten we wanneer glitches kunnen worden afgevuurd om te proberen de beveiligingen te omzeilen. Enkele van de bekende beveiligingsmethoden bij bankkaarten zijn:

1. Gebruik van een interne RC-klok voor veiligheidskritische bewerkingen, en overschakeling op externe CLK alleen voor nauwkeurige timing, bijvoorbeeld bij communicatie via de UART. Dit voorkomt klok-glitch aanvallen.
2. Het gebruik van zeer snelle ladingspompen op kleine interne (pF-bereik) voedingscondensatoren om een stabiele voedingsspanning te handhaven. Dit voorkomt spanningsglitch-aanvallen.
3. Het willekeurig veranderen van de halve perioden van de interne RC-klok (op basis van een interne echte toevalsgenerator, TRNG). Op die manier zullen de kritieke bewerkingen (zoals het controleren van de PIN-code) niet altijd op hetzelfde moment plaatsvinden, wat aanvallen bemoeilijkt.
4. Het toevoegen van willekeurige ruis aan de voedingsstroom. Dit voorkomt stroomanalyse-aanvallen (**figuur 10**).
5. Wanneer een PIN wordt gecontroleerd, wordt eerst de PIN try-teller in de EEPROM van de SmartCard verlaagd, wordt vervolgens de PIN gecontroleerd en wordt daarna de PIN try-teller weer verhoogd als de PIN in orde is. Oude kaarten schreven alleen naar EEPROM om de teller te verlagen bij een verkeerde PIN-invoer. Dit maakte brute force-aanvallen om de PIN-code te kraken mogelijk door snel de voeding uit te schakelen na elke verkeerde poging.
6. Gebruik van uiterst zorgvuldig gemaakte functies voor veiligheidskritische bewerkingen (hier is assembler-programmering nodig!), die altijd hetzelfde aantal CPU-klokcycli en (mogelijk) dezelfde hoeveelheid stroom verbruiken, ongeacht de inputvariabelen. Dit voorkomt timing- en stroomanalyse-aanvallen.
7. Het verlengen van een veiligheidskritische operatie, die normaal gesproken minder dan 1 ms duurt, tot bijvoorbeeld 200 ms. Een 'echt' signaal van 1 ms zit dus verborgen in 199 ms verwerking van 'garbage.'

Een moderne bankkaart niet-invasief uitschakelen (als dat al mogelijk is) vereist uitgebreide analyse van geregistreerde gegevens en lang priegelwerk. Lees het boek *Power Analysis Attacks* [5] voor uitgebreidere

en uiterst gedetailleerde informatie over voedingsanalyse-aanvallen.

Testen en leren

De Poor Man's ChipTweaker PMCT is geen hi-tech schakeling, maar kan worden gebruikt voor het uitproberen en leren van niet-invasieve aanvallen op HSM's. Ik hoop dat je dit artikel interessant vond, in ieder geval als een goed uitgangspunt. Het ontwerpen en aanvallen van single-chip HSM's is tegenwoordig voor alle betrokkenen erg moeilijk. Juist daarom blijft dit gebied open voor verder onderzoek.
Kraak ze!

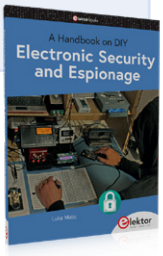
210462-03

Vragen of opmerkingen?

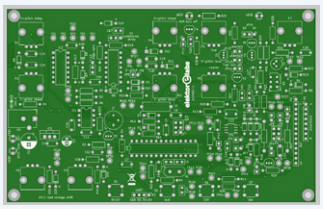
Hebt u technische vragen of opmerkingen naar aanleiding van dit artikel? Stuur een e-mail naar de auteur via luka.matic@fer.hr of naar de redactie van Elektor via redactie@elektor.com.

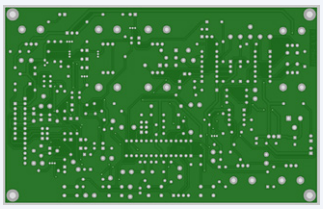
Gerelateerde producten

- **LabNation SmartScope USB Oscilloscope (SKU 17169)**
  www.elektor.nl/17169
- **Luka Matic, *Electronic Security and Espionage*, Elektor 2021 (SKU 19903)**
  www.elektor.nl/19903



ONDERDELENLIJST





Weerstanden:
R1,R29,R35 = 680 Ω
R2,R34 = 47 k
R3,R5,R6,R7,R9,R23,R25,R33,R38 = 10 k
R4,R8,R13 = 100 k
R10 = 1k5
R11,R12,R17,R22,R24,R26,R27,R32,R36 = 1 k
R14,R28,R30 = 4k7
R15 = 330 Ω
R16,R19 = 33 k
R18 = 2k2
R20* = 22 Ω
R21* = 10 k
R31,R37 = 100 Ω
R39,R40 = 3k3
P1,P2,P3,P5 = potmeter, 5 k, verticaal
P4 = potmeter, 10 k, verticaal
P6 = potmeter, 500 Ω, verticaal
P7 = potmeter, 2 k, verticaal
P8,P9 = potmeter, 25 k, verticaal

Condensatoren:
C1 = 100 µ/16 V, steek 3,5 mm
C2 = 10 µ/16 V, steek 2,5 mm
C3,C4,C7,C8,C9,C10,C11,C12,C20,C26,C28,C29 = 100 n, steek 5 mm
C5,C6,C17,C22,C25 = 22 p, steek 2,5 mm
C13,C24 = 100 pF, steek 2,5 mm
C14* = 10 n, steek 5 mm pitch
C27 = 10 n, steek 5 mm
C15*,C16* = 22 p, steek 5 mm
C18* = 33 p, steek 5 mm
C19* = 100 p, steek 5 mm
C21 = 220 p, steek 2,5 mm
C23 = 1 µ, steek 5 mm

Spoelen:
L1 = 10 µH

Halfgeleiders:
D1,D2,D3,D4,D5,D13,D14,D20,D21 = 1N4148
D11* = 1N4148
D6,D7,D8,D9,D10,D15,D16,D17,D18,D19 = BAT85
D12* = BAT85
IC1 = LM358
IC2 = 74HC00
IC3 = 74HC14
IC4 = LMV761, SOIC8
IC5 = 7805, TO220
IC6* = ATmega8A-PU
LED1,LED2 = LED, 3 mm, groen
LED3 = LED, 3 mm, geel
LED4 = LED, 3 mm, rood
T1 = BC141, TO39
T2 = BC557C
T3,T4,T5,T8,T9 = PN2369, TO92
T6 = BC547C
T7 = 2N2369, TO18

Diversen:
JP1...JP13,JP15 = 2-polige pinheader, raster 2,54 mm
JP14,S3,S4 = 3-polige pinheader, raster 2,54 mm
K1 = 8-polige pinheader, raster 2,54 mm
K2 = 14-polige pinheader, raster 2,54 mm
K3 = 4-polige pinheader, raster 2,54 mm
K4,K5 = 2-rijige 6-polige pinheader, raster 2,54 mm
K6 = voedingsconnector
S5 = schuifschakelaar (C&K JS202011CQN)
X1* = kristal 8 MHz

* monteer met busstrip voor gemakkelijk experimenteren

Funcard (bron: www.cellularcenter.it).

WEB LINKS

[1] Luka Matic, "Verzendbox met manipulatie-indicatie", Elektor mei/juni 2020: https://www.elektormagazine.nl/magazine/elektor-149/58607
[2] ChipWhisperers van NewAE Technology Inc.: https://www.newae.com/chipwhisperer
[3] Sergei P. Skorobogatov, "Copy Protection in Modern Microcontrollers": https://www.cl.cam.ac.uk/~sps32/mcu_lock.html
[4] Downloads bij dit artikel op Elektor Labs: https://www.elektormagazine.com/labs/poor-mans-chipwhisperer-or-a-smartcard-tweaker
[5] S. Mangard, E. Oswald, T. Popp, Power Analysis Attacks: Revealing the Secrets of Smart Cards, Springer, 2007: https://amzn.to/3RWBtuy

elektor TV

Elektor TV Shows

Elektor **Engineering Insights**

Elektor Industry Insights is een 'go-to' bron voor engineers en professionals die op de hoogte willen blijven over de wereld van de elektronica. Tijdens elke aflevering spreekt Elektor-redacteur Stuart Cording met experts uit de elektronica-industrie over technische uitdagingen en oplossingen.
www.elektormagazine.com/elektor-engineering-insights

Elektor LabTalk

Heb jij een passie voor DIY-elektronica, embedded systemen of algemene engineering theorie? Discussieer mee met het Elektor Labs team en onze redacteuren. Ze geven technische tips en bespreken toekomstige elektronicaprojecten en de inhoud van Elektor Magazine. En uiteraard beantwoorden ze al jouw vragen!
www.elektormagazine.com/elektor-lab-talk

elektor academy

Will jij jouw elektronicavaardigheden verbeteren? Dan is de Elektor Academy de 'place to be'. Onze expert Stuart Cording helpt u op weg via diverse cursussen over de meest uiteenlopende onderwerpen.
www.elektormagazine.com/elektor-academy

Blijf op de hoogte en volg Elektor TV
www.youtube.com/c/ElektorIM

elektor
design > share > earn