Bumping attacks:
the affordable way of obtaining chip secrets

Sergei Skorobogatov

http://www.cl.cam.ac.uk/~sps32 ***email: sps32@cam.ac.uk***

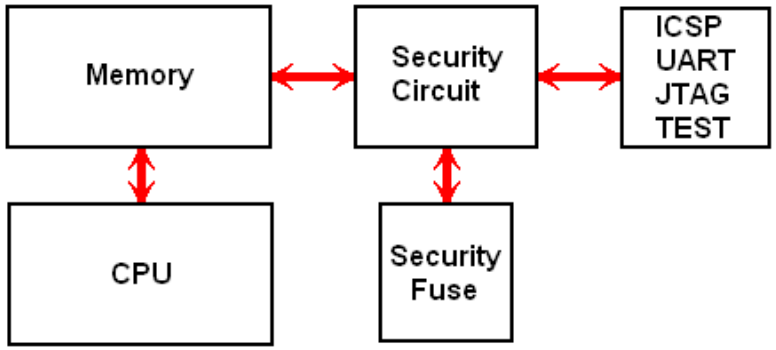
UNIVERSITY OF CAMBRIDGE
Computer Laboratory

Talk Outline

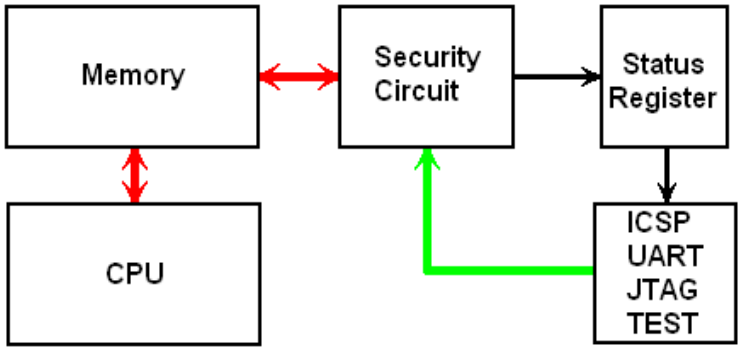
- Based on research work presented at CHES2010
  - Sergei Skorobogatov: Flash Memory 'Bumping' Attacks. Cryptographic Hardware and Embedded Systems Workshop (CHES-2010), August 2010, LNCS 6225, Springer-Verlag, ISBN 3-642-15030-6, pp 158-172
- Extended with partial reverse engineering of Actel FPGA
  - made non-invasive approach possible for the bumping attacks
- Countermeasures or proper security engineering
- Slides
  - http://www.cl.cam.ac.uk/~sps32/SG_talk_BA.pdf

Introduction: What is attacked?

- Data protection with integrity check
  - verify memory integrity without compromising confidentiality
  - How secure is the “No Readback” solution?



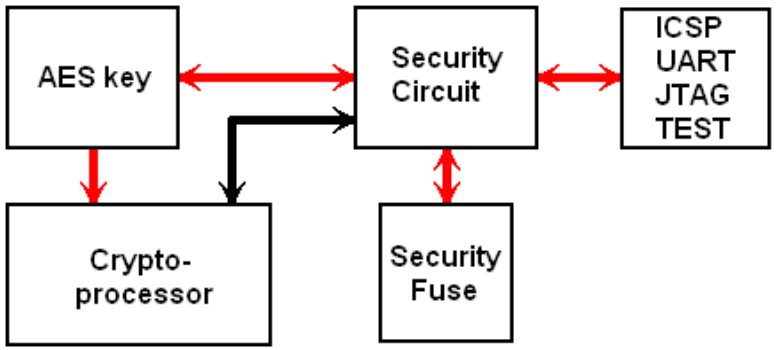
Readback access controlled by security fuse



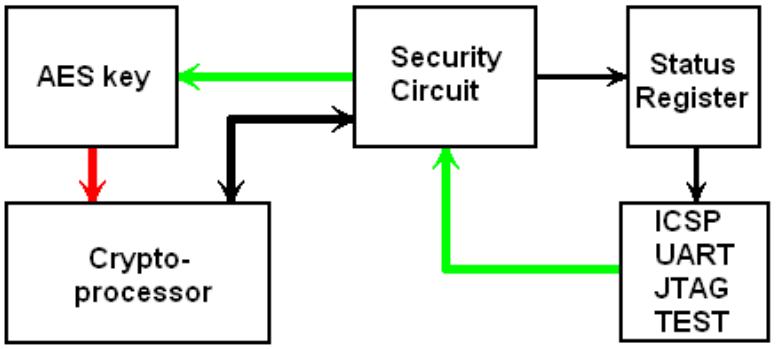
No Readback access
only secure verification

Introduction: What is attacked?

- Authentication using encryption
  - verify if a user knows the secret key by asking him to encrypt a message with his key
  - How secure is the 'No Readback' scheme against key extraction?



Readback access controlled by security fuse



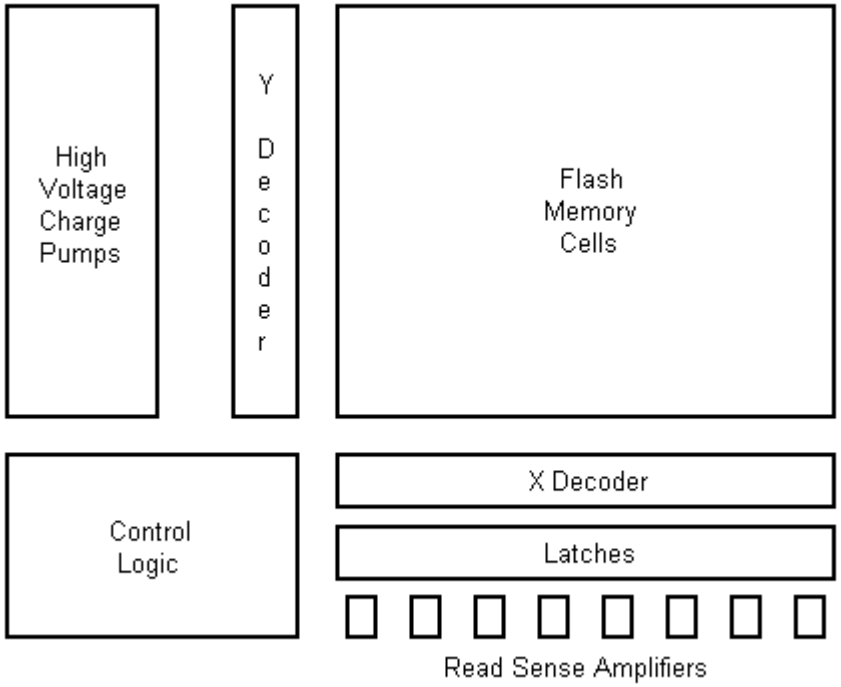
No Readback access
only secure verification

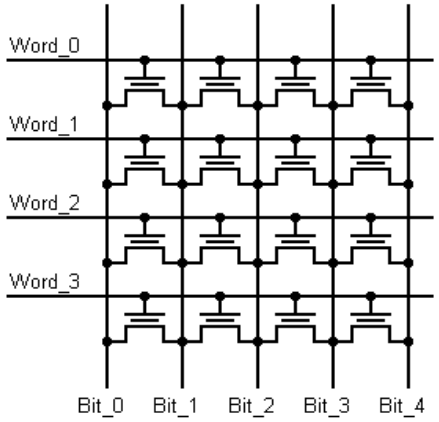
Introduction: Where is the key?

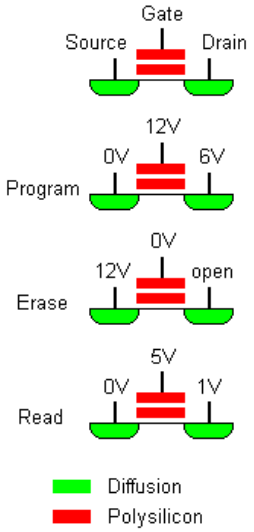
- Flash memory prevails
  - usually stores IP, sensitive data, passwords and encryption keys
  - widely used in microcontrollers, smartcards and some FPGAs
  - non-volatile (live at power-up) and reprogrammable, it can be OTP
  - low-power (longer battery life)
- How secure is Flash memory storage?
  - used in smartcards and secure memory chips, so it has to be secure
  - used in CPLDs by Xilinx and believed to be highly secure
  - used in secure FPGAs by Actel, marketed as “virtually unbreakable”
- Vulnerabilities of Flash memory found during my research
  - power glitching influence on data read from memory (Web2000)
  - optical fault injection changes data values (CHES2002)
  - laser scanning techniques reveal memory contents (PhD2004)
  - data remanence allows recovery of erased data (CHES2005)
  - optical emission analysis allows direct data recovery (FDTC2009)

Background: Flash memory

- Flash memory structure
  - high voltages required for operation
  - narrow data bus
  - dedicated control logic

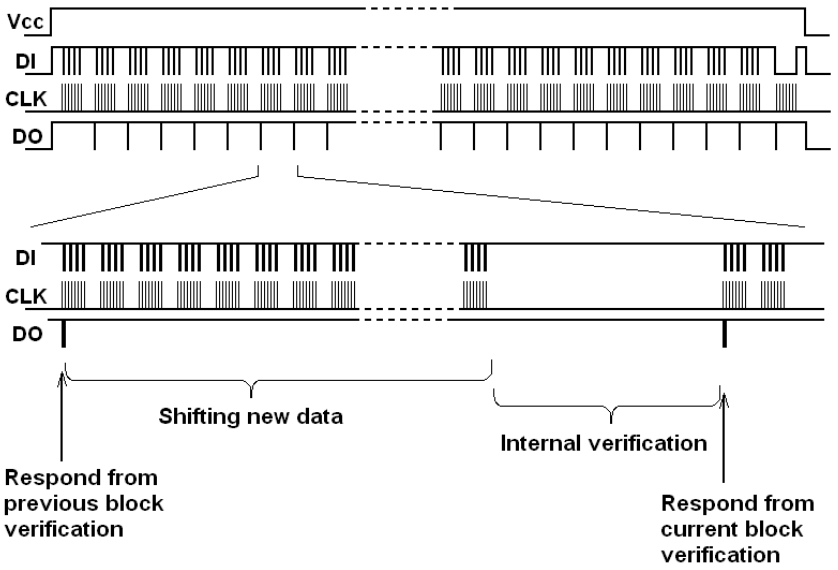


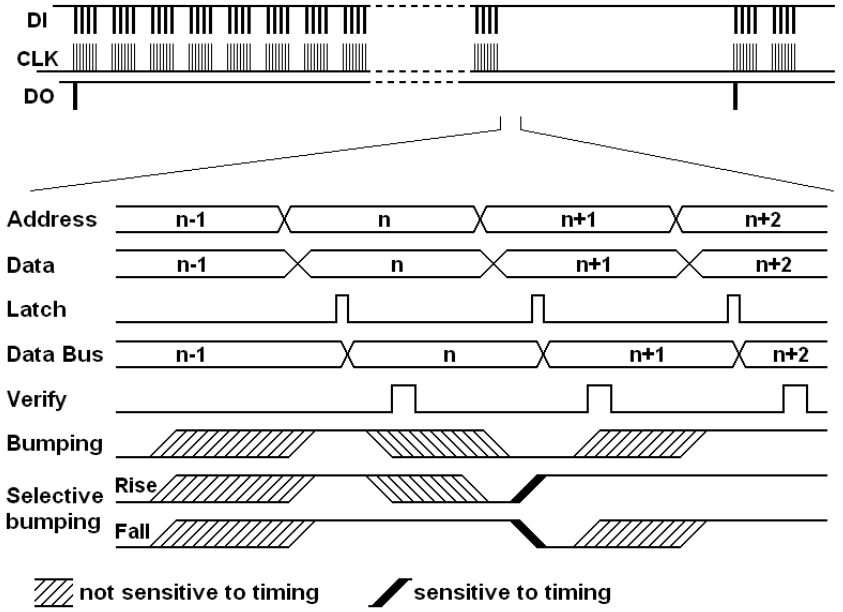




Background: Bumping attacks

- 'Bumping' is a certain type of physical attack on door locks
- Memory 'Bumping attacks' is a new class of fault injection attacks aimed at the internal integrity check procedure on-chip
  - 'bumping' is aimed at blocks of data down to bus width
  - 'selective bumping' is aimed at individual bits within the bus



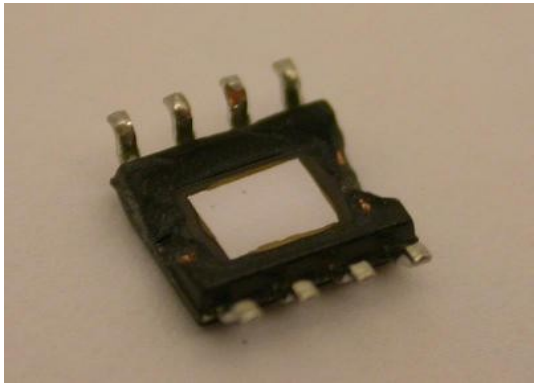


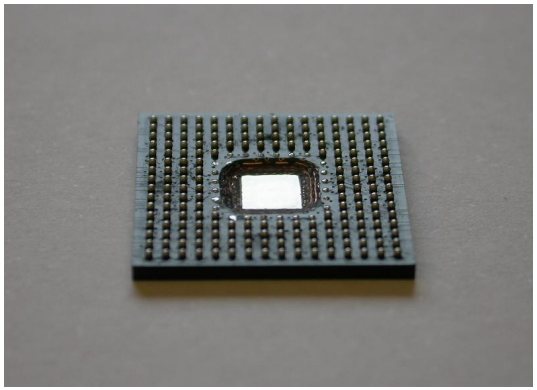
Attack methods

- Non-invasive attacks
  - observe or manipulate with the chip without any physical harm to it
  - low-cost: require relatively simple equipment and basic knowledge
  - time consuming and not always successful
- Invasive attacks
  - almost unlimited capabilities in extracting information and understanding chip functionality
  - expensive, requires a very sophisticated equipment and knowledge
  - less time consuming and straightforward for many devices
- Semi-invasive attacks
  - fill the gap between non-invasive and invasive types: direct access to the chip's surface is required but without any physical harm to it
  - moderate cost: some equipment can be easily built
  - higher success rate compared to non-invasive attacks
  - some are easily repeatable and relatively quick to set up

Experimental setup

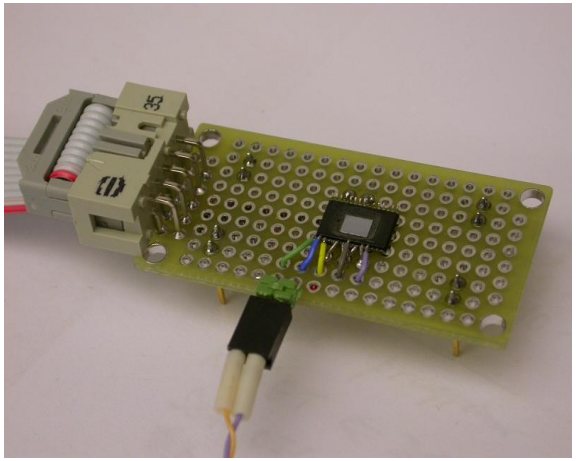
- Sample preparation for modern chips (<0.5µm and >2M)
  - only backside approach is effective
  - it is very simple and inexpensive
  - no chemicals are required

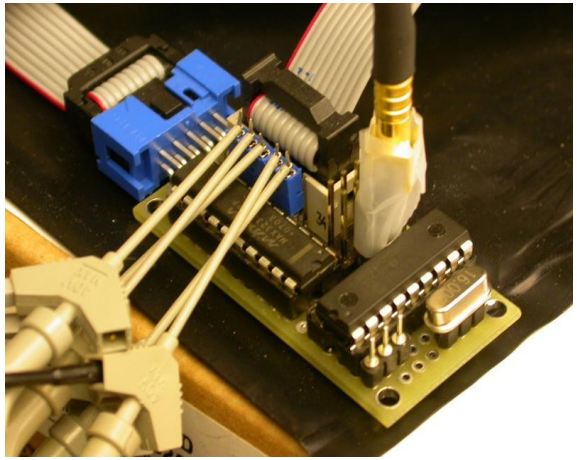


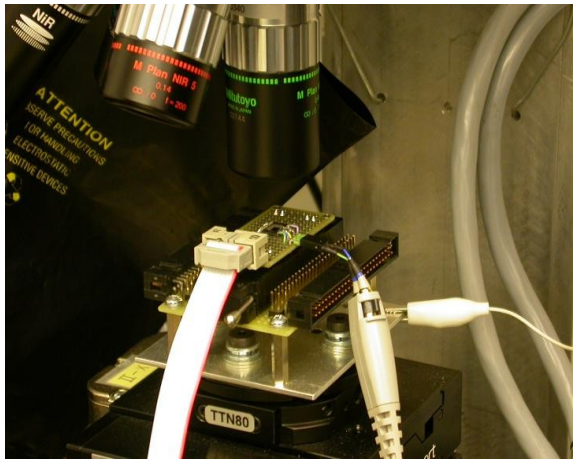


Experimental setup

- NEC 78K/0S μPD78F9116 microcontroller with 16kB Flash
  - memory access via bootloader for Erase, Write, Verify, Blank Check
  - 0.35μm process with 3 metal layers
- Optical fault injection attack
  - 1065nm laser diode module with output power up to 100mW
  - NIR objective lens with 20× magnification



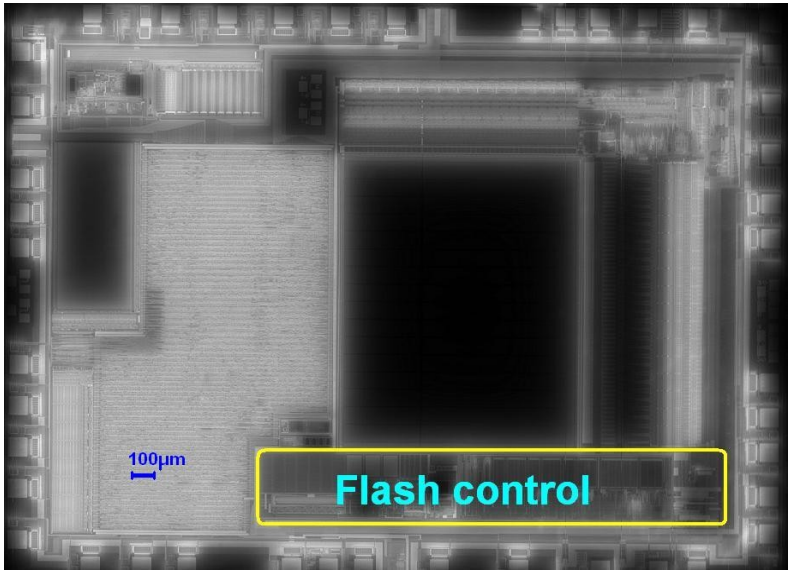


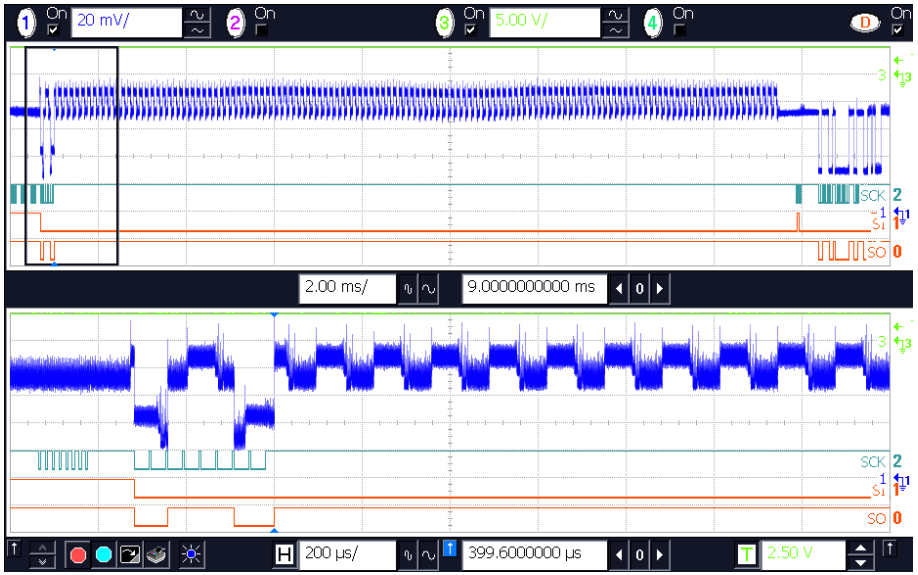


Results for bumping

- Locating Flash and active areas is easy (laser scanning)
- SPI interface for data transfer and SPA for timing analysis
- Memory matches all '0' when the laser is switched on
- Verification result is available only after all bytes are compared
- Data extraction time: 10 hours per block, or 2 months per chip

  2^7 attempts per byte, 128 bytes per block, 128 blocks, 2s per cycle



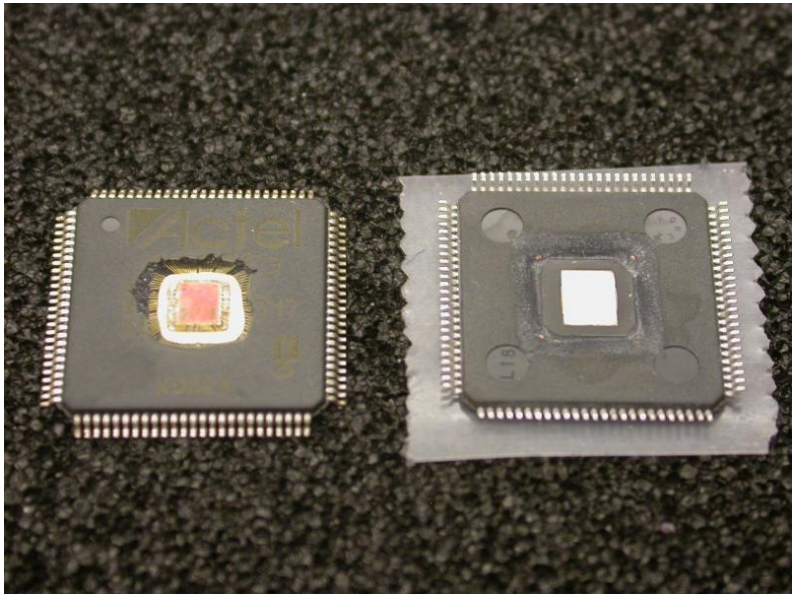


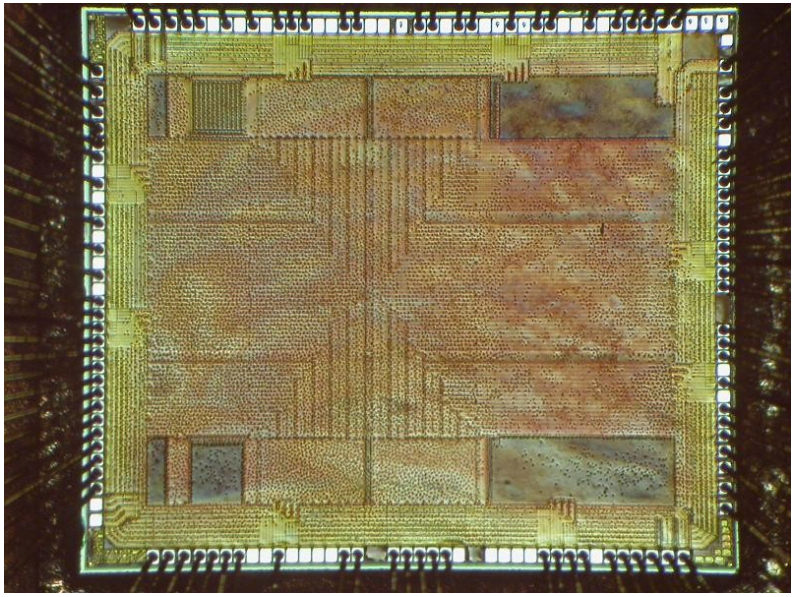
New challenge

- Actel® ProASIC3® 0.13µm, 7 metal layers, Flash FPGA
  - *"live at power-up, low-power, highly secure"; "impossible to copy"*
  - *"offer one of the highest levels of design security in the industry"*
  - *"unique in being reprogrammable and highly resistant to both invasive and noninvasive attacks"*
  - *"even without any security measures (such as FlashLock with AES), it is not possible to read back the programming data from a programmed device. Upon programming completion, the programming algorithm will reload the programming data into the device. The device will then use built-in circuitry to determine if it was programmed correctly"*
  - allows secure remote field updates with 128-bit AES-encrypted bitstream, AES authentication and MAC verification
  - other security measures: voltage monitors, internal charge pumps, asynchronous internal clock and lack of information about JTAG

Experimental setup

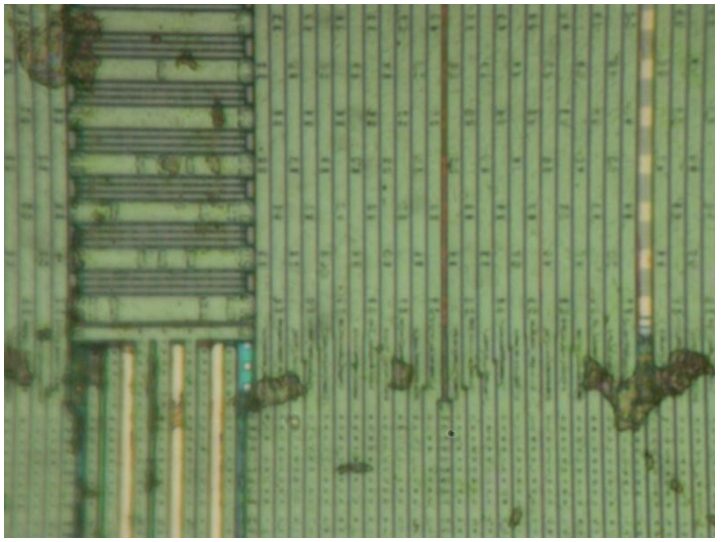
- Sample preparation of ProASIC3 FPGA: front and rear
  - the surface is covered with sticky polymer which needs to be removed for physical access to the surface
  - >99% of the surface is covered with supply grid or dummy fillers
  - backside: low-cost approach used – without any special treatment

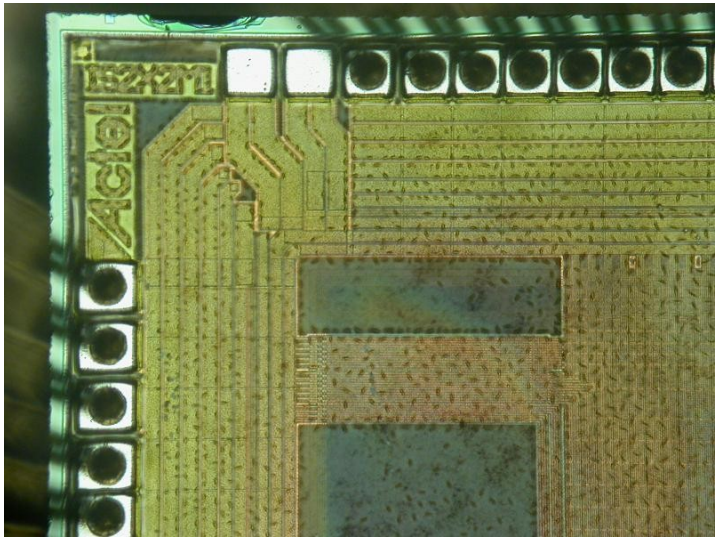


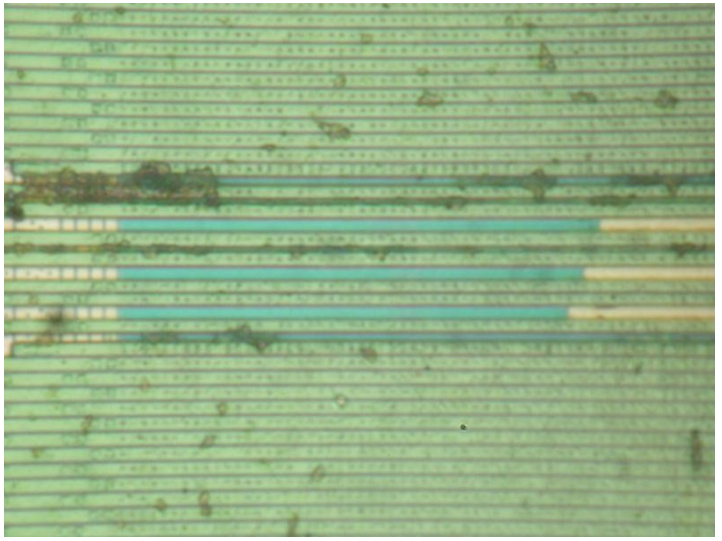


Experimental setup

- Sample preparation: front
  - only three top metal layers are visible at a most
  - full imaging will require de-layering and scanning electron microscopy
  - any invasive attacks will require sophisticated and expensive equipment

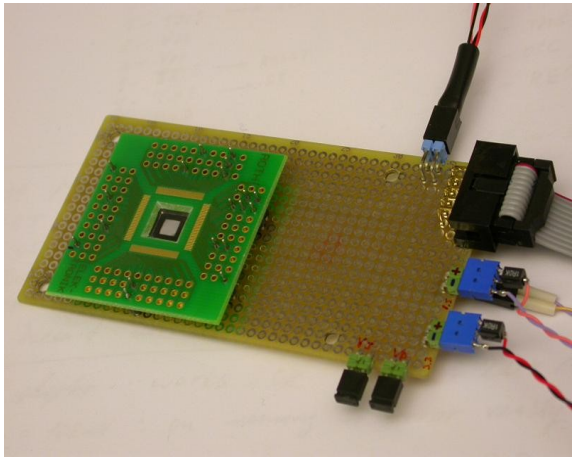


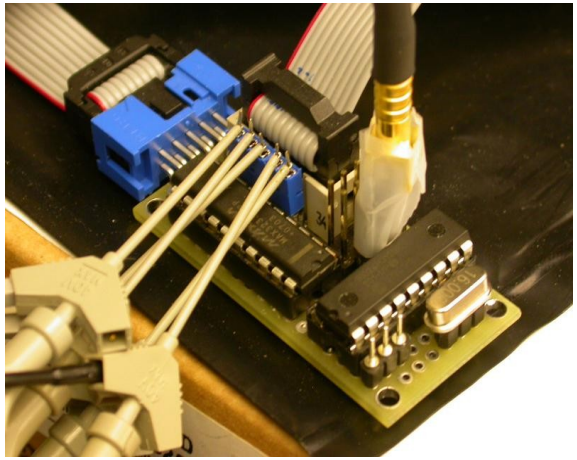


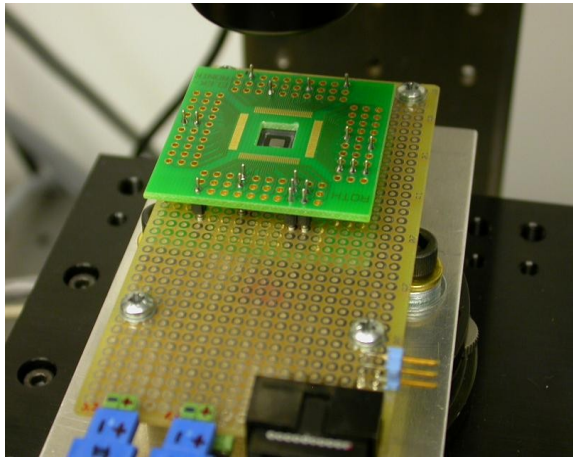


Experimental setup

- Actel ProASIC3 Flash-based A3P250 FPGA
  - limited information is available, but designs are loaded via JTAG
  - memory access via JTAG for Erase, Program and Verify operations
  - *"there is NO readback mechanism on PA3 devices"*
  - soon after introduction of optical fault attacks I warned Actel about possible outcomes for Flash technology, but they showed no interest
- Optical fault injection attack setup
  - chip on a test board under microscope with 20× and 1065nm laser



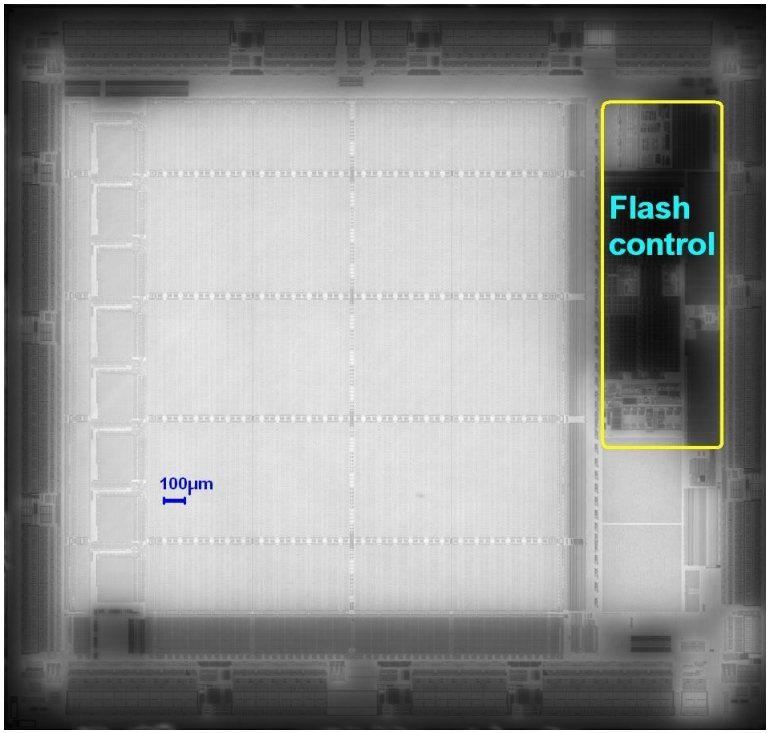


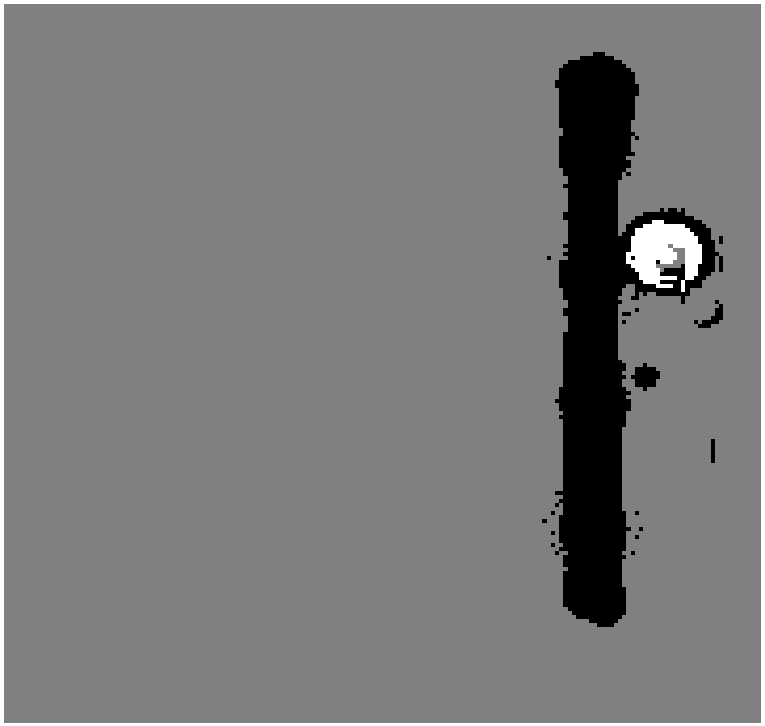


Results

- Locating Flash and active areas is easy via laser scanning
- JTAG interface was used for communication in Verify mode
- Sensitive locations were found with exhaustive search

  20μm grid: black – data corrupted, white – matching all '1'

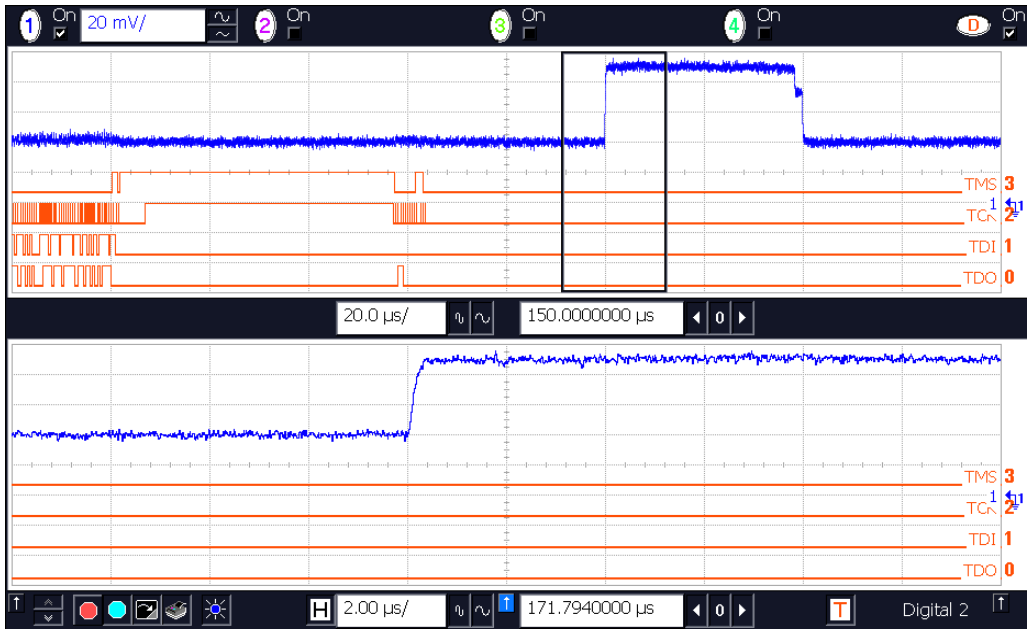




Results for bumping

- Using SPA for timing analysis: cannot detect data timing
- Verification result is available after each block of 832 bits
- 2300 blocks per array, 26 of 32-bit words per block
- Data extraction time: 18 years per block, 40000 years/chip

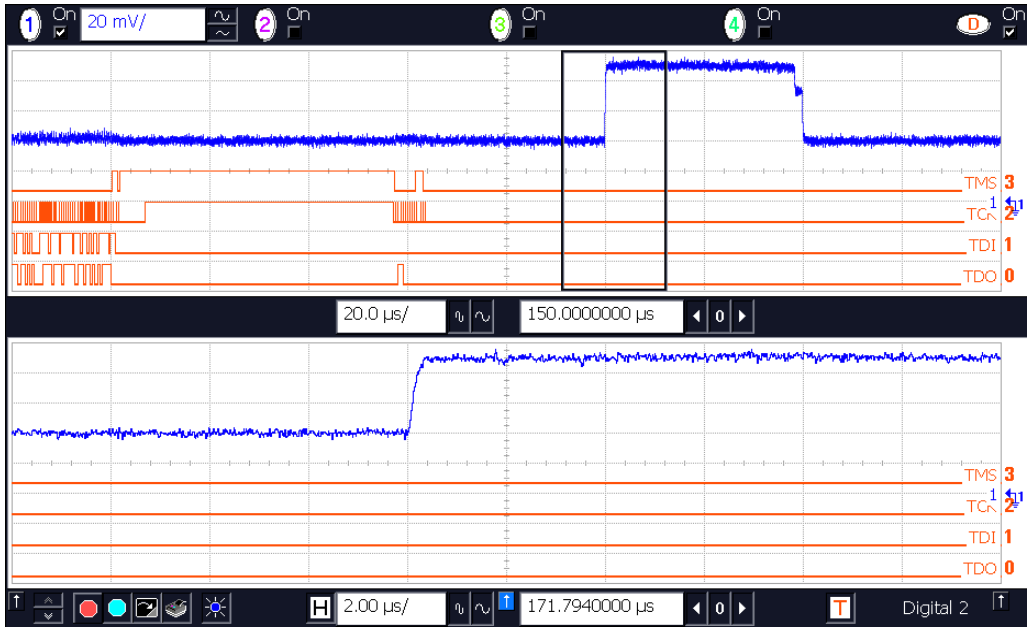
  2^{31} attempts per word, 26 words per block, 10ms per cycle



Results for selective bumping

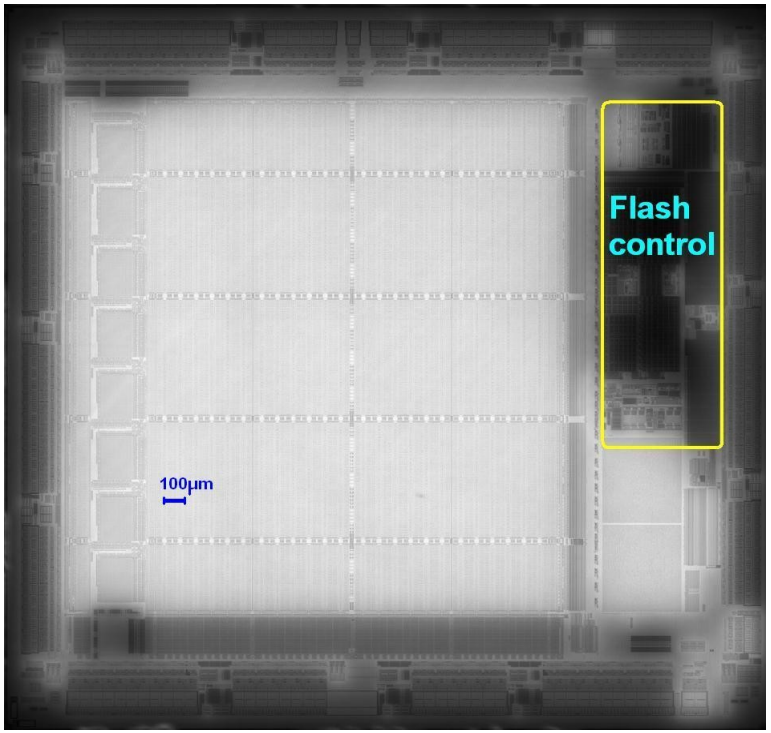
- Using SPA results as a time reference
  - block verification 40μs, 26 of 32-bit words per block, 1.5μs/word
- Laser switching time was adjusted in 25ns steps
  - searching for single '0' bit, then two '0' and so on until passed
- Data extraction time: 30 minutes per block, 50 days/chip

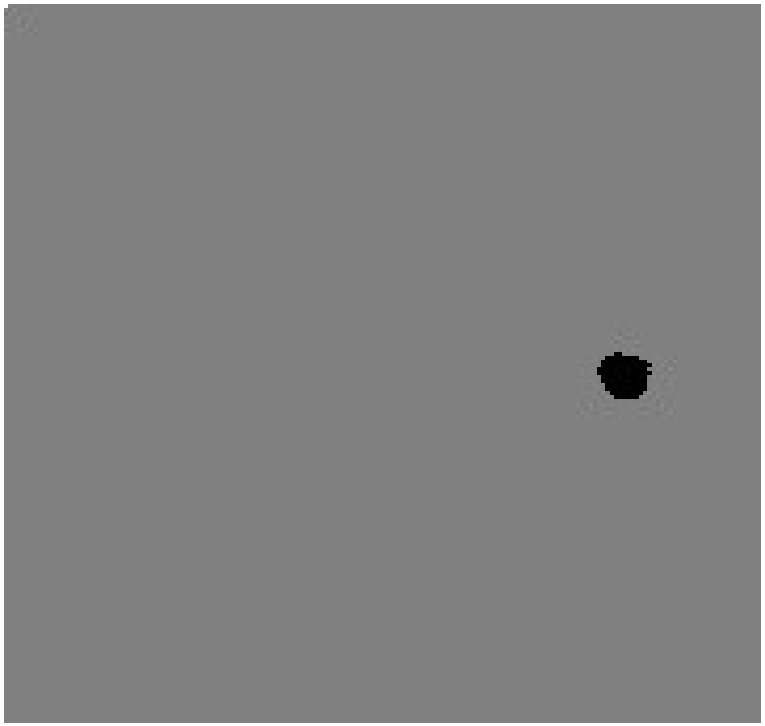
  2^{13} attempts per word, 26 words per block, 10ms per cycle



Results

- Trying to attack the AES key
- JTAG interface was run in AES authentication mode
- Sensitive locations were found with exhaustive search
  20μm grid: black – data corrupted, no white areas – failed



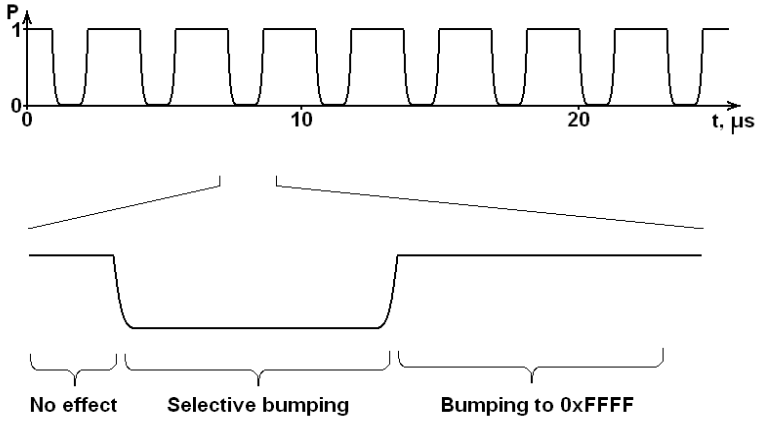


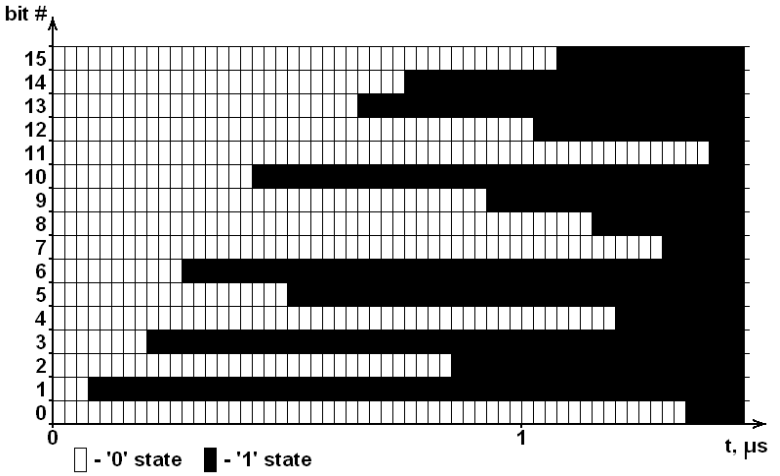
New challenge

- 90nm Secure ARM microcontroller with AES crypto-engine
  - secure memory for AES-128 key storage
  - permanent JTAG disable fuse
  - code is executed from the internal SRAM that can be loaded from AES-encrypted external NAND, SD or SPI Flash memory
  - supports AES-encrypted embedded Linux kernel
  - once activated the AES key is read protected and cannot be altered
  - other security measures are also in place

Experimental setup

- Analysis of the selective bumping phenomenon using the secure microcontroller with AES authentication
  - hardware setup was supplied by industrial sponsor
  - chip was supplied pre-programmed with a test AES key
  - glitching time was adjusted in 25ns steps
- Non-invasive power supply glitching attack was found
  - bumping: 2^{15} attempts per 16-bit word, 100ms cycle, 8 hours for AES key
  - selective bumping: 2^{7} attempts per 16-bit word, 2 minutes for AES key





Attack time on 128-bit block

- Without any improvements: brute force search
  - requires on average 2^{127} attempts
- Bumping: down to bus width
  - 8-bit bus: $2^7 \times 16 = 2^{11}$ attempts
  - 16-bit bus: $2^{15} \times 8 = 2^{18}$ attempts
  - 32-bit bus: $2^{31} \times 4 = 2^{33}$ attempts
- Selective bumping: down to single bit in limited steps
  - 8-bit bus: $(1+8+7+6+5+4+3+2+1)\times\frac{1}{2}\times16 \approx 2^8$ attempts
  - 16-bit bus: $(1+16+15+...+2+1)\times\frac{1}{2}\times8 \approx 2^9$ attempts
  - 32-bit bus: $(1+32+31+...+2+1)\times\frac{1}{2}\times4 \approx 2^{10}$ attempts
- In a real attack the complexity could be higher due to the granularity of the delay time and timing jitter
  - 32-bit bus: $(1+32+31+...+2+1)\times\frac{1}{2}\times4\times8\times4 \approx 2^{15}$ attempts

Limitations

- Slow process
  - depends on the implementation of data verification or authentication
- Precision timing is not necessary
  - slowly increase the delay until the effect is observed
- Selective bumping attacks have partial repeatability
  - individual bits within a memory row have different path lengths
  - slight variation between memory rows due to transistors parameters
- Fault attacks can be carried out with glitching or optically
  - optical attacks on modern chips require backside approach
- Precise positioning for optical attacks is not necessary, but a stable optical bench is required for a long run attack
- Security with no readback is not the only one in ProASIC3
  - passkey access protection, AES encryption, security fuses

Improvements

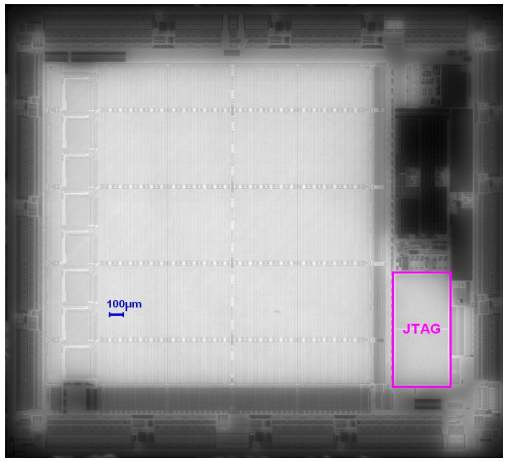
- Moving away from semi-invasive attacks toward using non-invasive attacks like in the last example with AES key extraction from the secure microcontroller
  - easier to setup for deep-submicron chips
  - faster to get the result
  - pose larger threat to the hardware security
- One approach is to use data remanence effect to help with bumping through threshold voltage adjustment
  - S. Skorobogatov: Data Remanence in Flash Memory Devices, CHES-2005, LNCS 3659, pp.339–353
- In Actel ProASIC3 FPGAs V_{CC} core supply voltage does not have enough influence on V_{TH} of the Flash cells, hence, need to somehow influence the read sense amplifiers

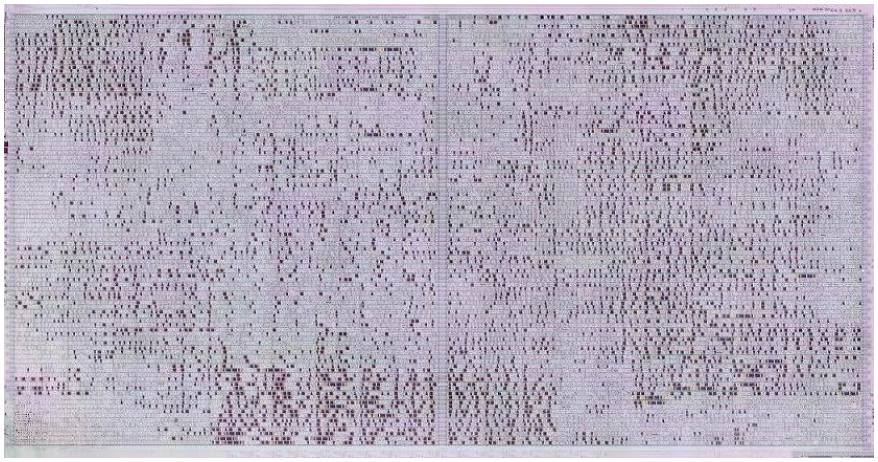
New challenge

- Non-invasive attack on Actel® ProASIC3® Flash FPGA
  - *"unique in being reprogrammable and highly resistant to both invasive and noninvasive attacks"*
  - *"on-board security mechanisms prevent access to the programming information from noninvasive attacks"*
  - *"special security keys are hidden throughout the fabric of the device, preventing internal probing and overwriting. They are located such that they cannot be accessed or bypassed without destroying the rest of the device, making both invasive and more subtle noninvasive attacks ineffective"*
  - other security measures: voltage monitors, internal charge pumps, asynchronous internal clock and lack of information about JTAG
- Mission Impossible 1: bypass multiple security protection
  - gain low-level control over the internal Flash hardware control logic and interfere with read sense amplifiers to influence V_{TH}-V_{REF}

Ways to approach

- Ask Actel if the chip has any backdoors or special features
  - even under the most NDA Actel would not admit the device has any backdoor access, even if there were such
- Straightforward invasive reverse engineering (40k gates)
  - open up the chip and remove layer by layer using deprocessing technique
  - take high-resolution digital photos and combine them into the layout map
  - create transistor level netlist of the device and convert it into gates level
  - organise gates into functional units and groups
  - simulate the whole system and find hidden functions and bugs
  - 6 to 12 months to extract the design with 300k to 2M GBP cost
  - 3 to 12 months to analyse the data



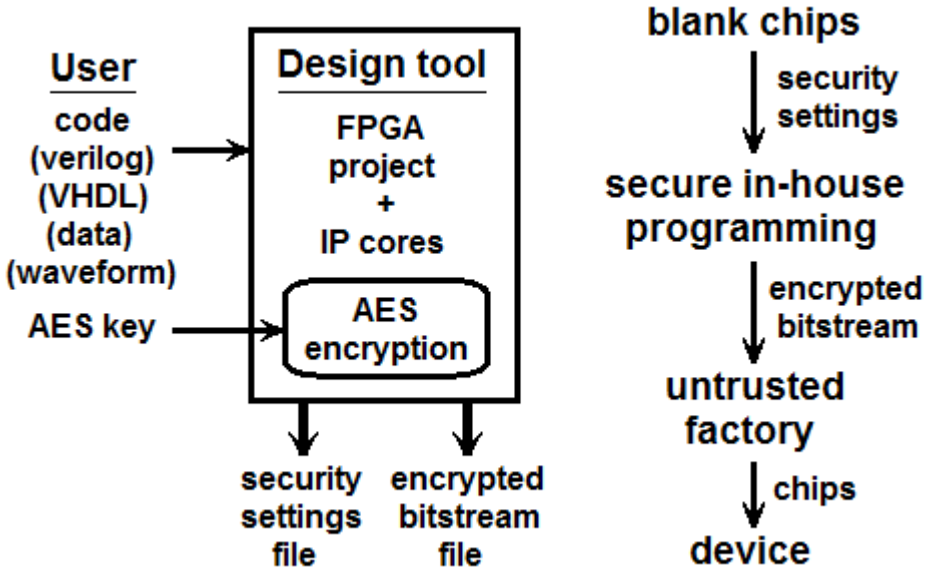


Ways to approach

- Do a bit of research
- Google it for code examples, disclosed information, patents
  - programming files with security settings
  - hint on V_{TH} compensation for RT devices
- Use development tools to generate programming files, use them and eavesdrop on JTAG communication; it is simpler
  - STAPL high-level language is used which is self-explanatory
- Company website and distributors for clues on the security
  - release notes and product descriptions mention *"dual-key security"*
  - *"board with a dual-key M1AFS600 device"*
- Why is the '**dual-key security'** is not mentioned in any Actel datasheets, press releases and white papers???

Secure AES-128 update in Actel FPGA

- Designed to prevent IP theft, cloning and overbuilding
- A3P600 vs M1A3P600 (ProASIC3 FPGA family)
  - if certain vendor IP cores are used (Cortex-M1) – no user protection
  - user AES DMK is used for Actel IP core protection (DMK = M1 key)



Dual-key security in Actel FPGA

- What problem does the dual-key security solve?
  - IP cores loyalty control without compromising user security
  - *"The system enables application development with the ARM Cortex-M1 and/or with your own optional AES key (owing to dual-key feature) in mixed-signal M1-enabled Fusion devices"*
- AFS600 vs M1AFS600 (Fusion mixed-signal FPGA family)
  - user AES DMK protects user's IP and 2nd key is for the vendor IP
  - when protection for both user IP and vendor IP are required then AES Key = H(user key, vendor key), H – secure hash function
  - in M1AFS600 the 2nd key = M1 key, what is the 2nd key in AFS600?

How the AES key can be attacked?

- Invasive attacks (expensive)
  - partial reverse engineering followed by microprobing
- Semi-invasive attacks (affordable)
  - optical fault injection attack (Skorobogatov, Anderson CHES2002)
  - optical emission analysis (Skorobogatov FDTC2009)
- Non-invasive attacks (simple)
  - side-channel attacks such as SPA, DPA, CPA, EMA, DEMA
  - poor signal-to-noise ratio of about −15dB due to low-power operation and multiple sources of noise (clocks, pumps, acquisition)

How long does it take to get the AES key?

- Invasive attacks (microprobing)
  - **1 day** with FIB and probing station
- Semi-invasive attacks (side-channel and fault attacks)
  - **1 week/1 hour** with optical emission analysis (FDTC2009)
  - **1 hour** with optical fault injection attack (CHES2002)
- Non-invasive attacks (side-channel attacks)
  - **1 day** with low-cost DPA setup: resistor in V_{CC} core supply line, oscilloscope with active probe and PC with MatLab software
  - **1 hour/10 minutes** with commercial DPA tools (DPA Workstation from Cryptography Research or Inspector SCA from Riscure)
  - **1 second** with QVL board using special SCA sensor from QVL
  - **0.01 second** with Pandora tester using breakthrough approach to power analysis technique from QVL

Quest for the factory secret master key

- Reverse engineering of the STAPL file for M1AFS600
  source: Google search that revealed Cortex-M1 Fusion Kit soft CD

```
PROCEDURE VERIFY_ID_DMK USES GV,DO_EXIT,INIT_AES;
   IRSTOP IRPAUSE;
   DRSTOP DRPAUSE;
   IRSCAN 8, $ac;
   DRSCAN 3, $1;
   WAIT IDLE, 1 CYCLES;
   CALL INIT_AES;
   IRSTOP IRPAUSE;
   DRSTOP DRPAUSE;
   IRSCAN 8, $0a;
   DRSCAN 128, $e137623a2eeee91126015f3f73664945;
   WAIT IDLE, 3 CYCLES;
   WAIT IDLE, 256 USEC;
   DRSCAN 128, $00000000000000000000000000000000, CAPTURE BUFF128[],COMPARE
$c00000000000000000000000000000000,$c00000000000000000000000000000000,PASS;
   IF ( ! (BUFF128[127]==0) ) THEN GOTO M7VERDONE;
   STATUS = -31;
   PRINT "Failed to verify AES Sec.";
   CALL DO_EXIT;
   M7VERDONE:
   IF ( ! ( (BUFF128[126]==0)||(BM7DEVICE==0)) ) THEN GOTO MXIDOK;
   IF ( ! ( (BUFF128[126]==1)&&(BM7DEVICE==0)) ) THEN GOTO LDETECTM1;
   STATUS = -32;
   PRINT "Failed to verify IDCODE.";
   PRINT "M7 Device detected.";
   CALL DO_EXIT;
   LDETECTM1:
   IF ( ! (BUFF128[126]==0) ) THEN GOTO Label_80;
   IRSTOP IRPAUSE;
   DRSTOP DRPAUSE;
   IRSCAN 8, $0a;
   DRSCAN 128, $acdd6548ccb488863e291eb18fe95077;
   WAIT IDLE, 3 CYCLES;
   WAIT IDLE, 256 USEC;
   DRSCAN 128, $00000000000000000000000000000000, CAPTURE BUFF128[],COMPARE
$c00000000000000000000000000000000,$c00000000000000000000000000000000,PASS;
   IF ( ! (BUFF128[127]==0) ) THEN GOTO M1VERDONE;
   STATUS = -31;
   PRINT "Failed to verify AES Sec.";
   CALL DO_EXIT;
   M1VERDONE:
   BOOLEAN BTMPBUFFBIT126 = BUFF128[126];
   IF ( ! ( (BTMPBUFFBIT126==1)&&(BM1DEVICE==0)) ) THEN GOTO REGDEV;
   STATUS = -32;
   PRINT "Failed to verify IDCODE.";
   PRINT "M1 Device detected.";
   CALL DO_EXIT;
```

Select AES mode (0 - user key, 1 - IP core key, 2 - dual key)

Calculate round keys

Authenticate M7 device as $\{00...00\}_{K=M7}$

Authentication passed for M7 device

Authenticate M1 device as $\{00...00\}_{K=M1}$

Authentication passed for M1 device

```
   REGDEV:
   IF ( ! (BTMPBUFFBIT126==0) ) THEN GOTO Label_77;
   IRSTOP IRPAUSE;
   DRSTOP DRPAUSE;
   IRSCAN 8, $0a;
   DRSCAN 128, $fdffe775fbb073c640529c6443500086;
   WAIT IDLE, 3 CYCLES;
   WAIT IDLE, 256 USEC;
   DRSCAN 128, $00000000000000000000000000000000, CAPTURE BUFF128[],COMPARE
$c00000000000000000000000000000000,$c00000000000000000000000000000000,PASS;
   IF ( ! (BUFF128[127]==0) ) THEN GOTO MCVERDONE;
   STATUS = -31;
   PRINT "Failed to verify AES Sec.";
   CALL DO_EXIT;
   MCVERDONE:
   IF ( ! (BUFF128[126]==0) ) THEN GOTO REGDEV2;
   STATUS = -32;
   PRINT "Failed to verify IDCODE.";
   PRINT "Core enabled device detected.";
   CALL DO_EXIT;
   REGDEV2:
   LABEL_SEPARATOR = 0;
   Label_77:
   IF ( ! ( (BTMPBUFFBIT126==0)&&(BM7DEVICE==1)) ) THEN GOTO Label_78;
   STATUS = -32;
   PRINT "Failed to verify IDCODE.";
   PRINT "The Target is not an M7 Device.";
   CALL DO_EXIT;
   Label_78:
   IF ( ! ( (BTMPBUFFBIT126==0)&&(BM1DEVICE==1)) ) THEN GOTO Label_79;
   STATUS = -32;
   PRINT "Failed to verify IDCODE.";
   PRINT "The Target is not an M1 Device.";
   CALL DO_EXIT;
   Label_79:
   LABEL_SEPARATOR = 0;
   Label_80:
   LABEL_SEPARATOR = 0;
   MXIDOK:
   LABEL_SEPARATOR = 0;
ENDPROC;
```

Authenticate ?? device as $\{00...00\}_{K=??}$

Authentication passed for core-enabled device

==> normal device has K=??

AES key extraction

M7: e137623a2eeee91126015f3f73664945 = $\{00...00\}_{K=4D\$0B}$

M1: acdd6548ccb488863e291eb18fe95077 = $\{00...00\}_{K=81\$74}$

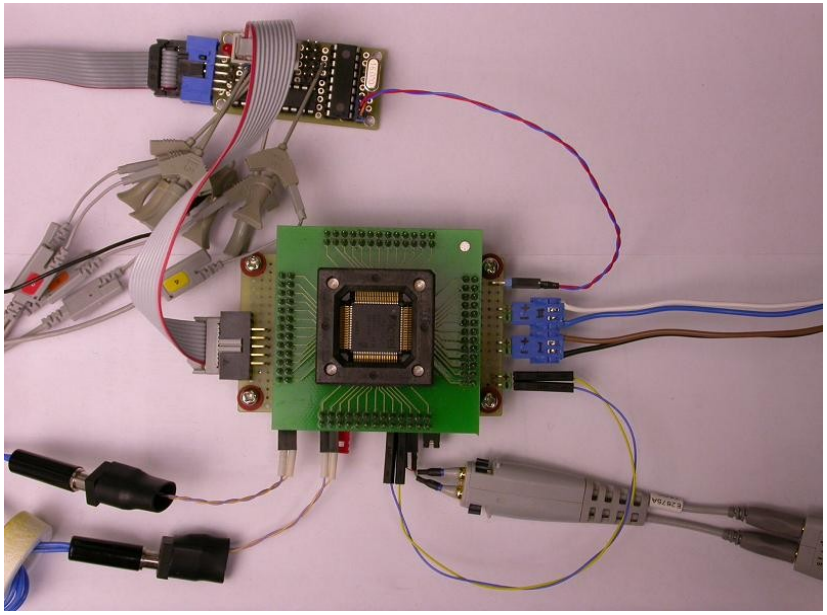
??: fdffe775fbb073c640529c6443500086 = $\{00...00\}_{K=09\$14}$

Results

- What can be done if the factory secret master key is known?
  - turn some ROM areas into reprogrammable Flash areas
  - reprogram low-level features
  - access shadow areas
  - access hidden JTAG registers
  - find the JTAG registers responsible for controlling read sense amplifiers, such that V_{REF} can be adjusted
- Actel's big security mistake
  - all Actel 3rd generation Flash FPGA devices (ProASIC3, ProASIC3L, ProASIC3 nano, Igloo, Igloo plus, Igloo nano, Fusion, SmartFusion) share the same factory secret master key
  - thanks to irresponsible corporate security strategy many Flash FPGA devices can now be manipulated
- Do we really have to go that long way to find the factory key?
  - YES, because it is somewhat million times harder to break the factory key than the AES key, thanks to side-channel leakages

Experimental setup

- Non-invasive bumping attack on Actel ProASIC3 Flash-based A3P250 FPGA
  - memory access via JTAG for Erase, Program and Verify operations
  - *"there is NO readback mechanism on PA3 devices"*
  - the secret JTAG registers set V_{REF} close to V_{TH} of the Flash cells
  - the test board is glitching V_{CC} to influence V_{TH} of the Flash cells



Results for bumping

- Using SPA results as a time reference
- Verification result is available after each block of 832 bits
- 2300 blocks per array, 26 of 32-bit words per block
- Two ways of approaching
  - set V_{REF}<min(V_{TH}) to flip all bits to ‘1’
  - set V_{REF}>max(V_{TH}) to flip all bits to ‘0’
- Power glitching of V_{CC} for the duration of N words and search for matching value
- Change V_{REF} and repeat V_{CC} glitching until all bits are found
- Number of bits changed at a time: from 1 to 4
- Data extraction time: 5 days per block, 30 years/chip

  2^{21} attempts per word, 26 words per block, 10ms per cycle

Results for selective bumping

- Using SPA results as a time reference
  - block verification 40µs, 26 of 32-bit words per block, 1.5µs/word
- Set V_{REF}<min(V_{TH}) to flip all bits to ‘1’
- Glitching V_{CC} at the time when the word value is latched into internal register and adjusting the timing in 25ns steps
  - searching for single '0‘ bit, then two '0' and so on until passed
- Data extraction time: 20 minutes per block, 30 days/chip

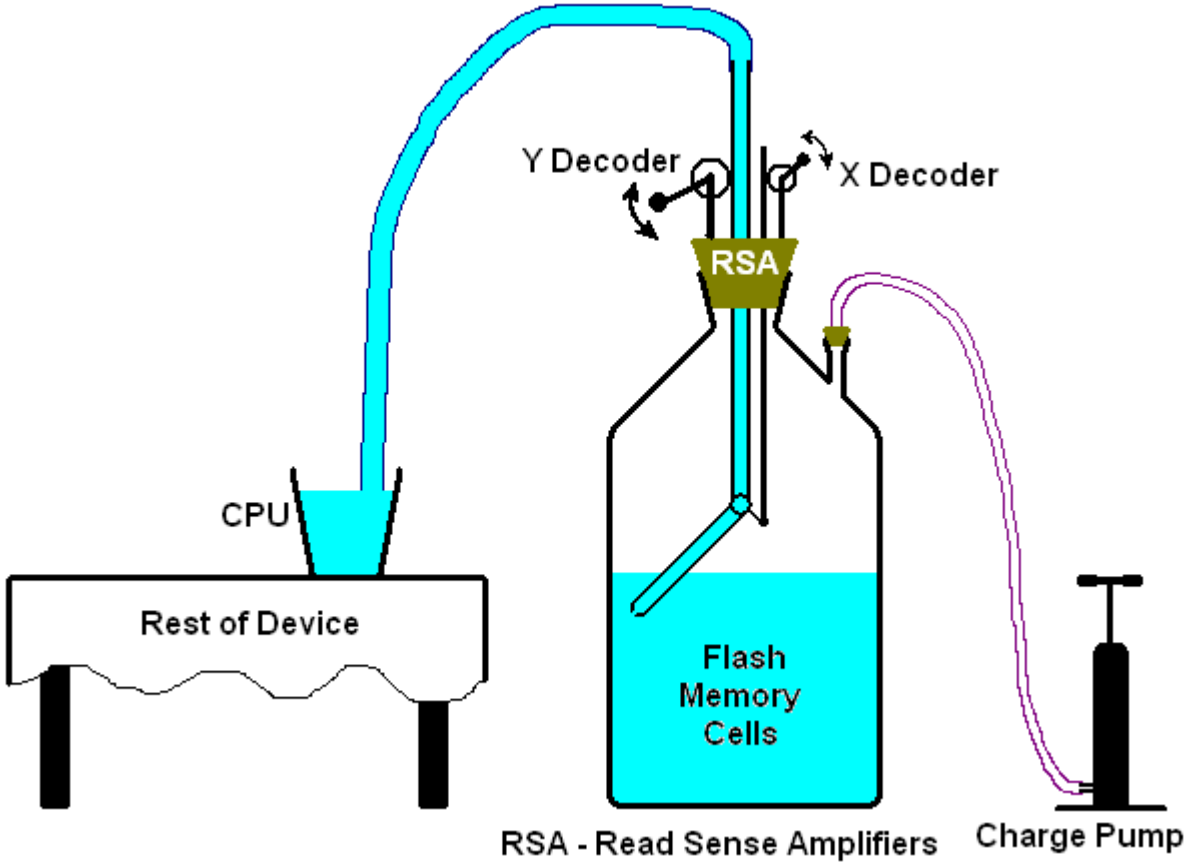
  2^{12} attempts per word, 26 words per block, 10ms per cycle

Countermeasures

- Encryption and redundancy check make analysis harder but not impossible
- Asynchronous circuits could make the attack more problematic as bumping requires predictable timing
- Dummy cycles will pose certain challenges to the attacker
- To develop adequate protection you must know how your device was attacked and compromised
- Never use your factory secret master key for authentication
- Never use the same master key in all of your products
- Use strong enough keys as a passwords – be creative, some devices have HEXspeak as a part of the key/password
  - DEADBEEF, BABE, BAD, 999, BEEF, A11
- Understanding the core of a problem is vital

Why Flash memory fails?

- Flash memory in a nutshell – for better understanding
  - can you see the bottleneck(s)?



Future work

- Data remanence analysis of embedded Flash in modern chips
- Testing other chips for strength against firmware and secret key extraction
- Mission Impossible 2: recover the erased data
  - according to Actel it is *"virtually impossible"* to extract the information from Actel ProASIC3 Flash-based FPGA
  - how about recovering the information after it has been erased?
  - quite common situation if you have overproduction of your highly secure designs and then decide to switch to another product; if you have pre-programmed secure chips left you might erase them into the initial state and sell on the market; that means those chips will no longer have any security fuses activated, but just 'No readback' feature and data remanence within the Flash memory

Conclusions

- Bumping attacks are dangerous and can compromise the security in chips – evaluation and protection is necessary
- Backside approach helps in modern chips, it is simple to do and does not require expensive optics and precise positioning
- Bumping attacks can be used for partial reverse engineering to understand internal data paths and chip structure
- The hardware security protection in Actel ProASIC3 FPGAs is under serious threat due to unforeseen problems in the corporate security strategy of the management team
- Access path to shadow hardware features brings capability of making ProASIC3 chips more robust and serve security critical applications for the next few years
- Embedded memory is more secure than encrypted external memory storage, and encrypted bitstream is even less secure

Conclusions

- It might be OK to have backdoors and trojans in highly secure devices, but they should be kept secret and never used to boost the existing security measures
- Is it OK if your products are used for military, space, avionics, medical, industrial control and other security critical applications?
- Odd stuff: ProASIC3 security problems were reported at CHES2010, one month later Actel was rushed into sale by the Board of Directors without consultations with shareholders, and on November 2nd 2010 Actel Corporation ended its existence

Conclusions

- If you do not want to get screwed talk to experts in academia; there are solutions for increasing the security of chips