University of BRISTOL

Sep.17-19 2018

Modeling M0 leakage generically

How different can leakages be among different M0 devices?

Si Gao

University of Bristol

Outline

Side Channel Analysis

SCA

- Attacks based on information leakage (timing, power consumption, electromagnetic emission, etc.)
- Recover the secret key potentially within a few minutes (1 — several million traces)

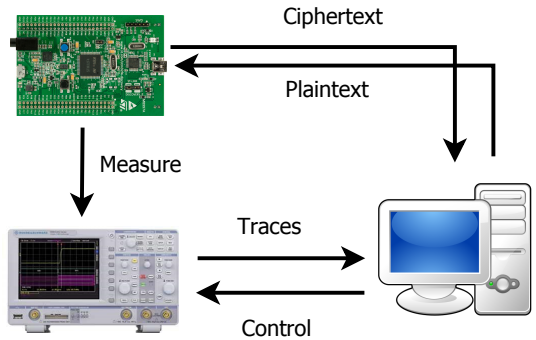


Figure: Side Channel Analysis

SCA on ARM processors

Popular target (good leakage/widely deployed). Interesting attacks are:

- Timing
  - Unbalanced branches (RSA [Koc96; BB05], DSA[Koc96], ElGamal [Koc96], etc.)/ branch prediction (Spectre [Koc+18])
  - Cache attack (data [Pag02; Ber05; TOS10]/instruction [Aci07])
  - Early-Terminating Multiplications [Gro+09]
- Power consumption/electromagnetic emission
  - ARM Cortex-A8: DPA on 1GHz bit-sliced AES [Bal+15]
  - ARM7TDMI: EMA on 40 MHz AES/ECC [GHT05], SPA on 832MHz AES/ECC [Nak+14]

Security Evaluation for power analysis

Countermeasures for power/EM analysis: tricky, prone to mistakes

Tools for Early Detection of Power Leakage

Require power simulators that may be at the level of ...

- Gate-level netlist: way too complex
- High level source code: the compiler may rearrange code
- Binary code/Assembly level/post compilation
  - HW/HD model: sufficient?
  - User defined model: leave the nasty part to users
  - Profiling model: Meet ELMO!

Meet ELMO

ELMO: **Emulator for power Leakages for the M0**

- Target: ARM Cortex M0
- Thumbulator + profiling power model (carefully derived from a **real** M0 core)
- Input: Binary Code
- Leakage detection with Fix v.s. Random test

Meet ELMO

Leakage detection: ELMO v.s. Real Traces

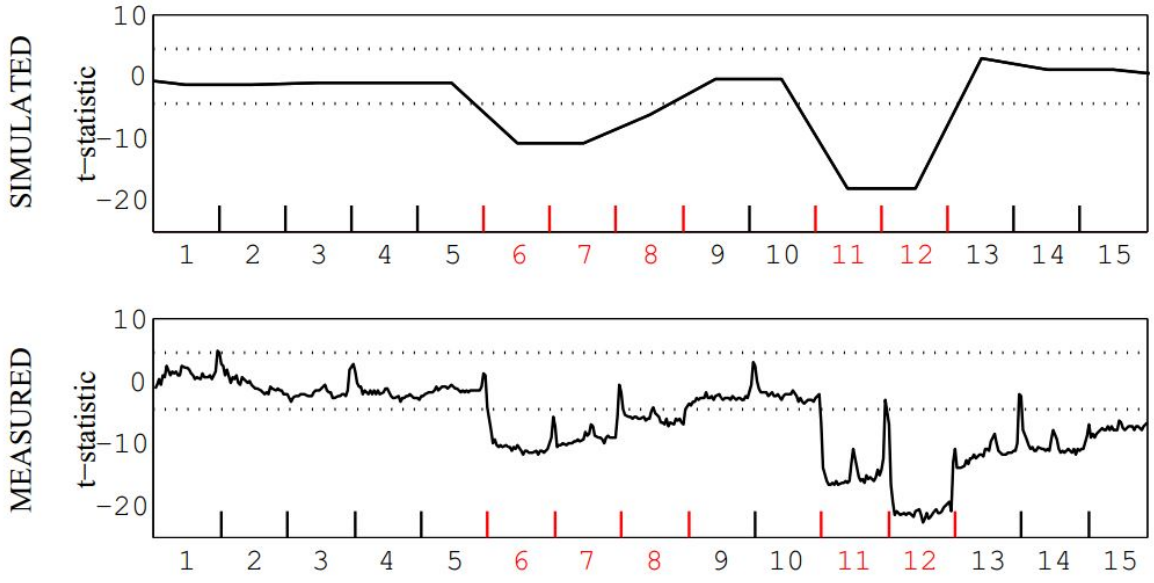


Figure: Fix v.s. Random T-Test for masked ShiftRows

ELMO captures the leaks as they occur also in the real traces.

Meet ELMO

Reasoning the detected leakage

Cycle No.	Address	Machine Code	Assembly Code
1-2	0x08000206	0x684C	ldr r4,[r1,#0x4]
3	0x08000208	0x41EC	ror r4,r5
4-5	0x0800020A	0x604C	str r4,[r1,#0x4]
6-7	0x0800020C	0x688C	ldr r4,[r1,#0x8]
8	0x0800020E	0x41F4	ror r4,r6
9-10	0x08000210	0x608C	str r4,[r1,#0x8]
11-12	0x08000212	0x68CC	ldr r4,[r1,#0xC
13	0x08000214	0x41FC	ror r4,r7
14-15	0x08000216	0x60CC	str r4,[r1,#0xC]

Figure: Thumb assembly implementation of masked ShiftRows

Beyond ELMO...

Big unresolved question: are the profiles that were derived from an STM32F0, suitable for other Cortex M0 cores?

- Manufacturer-specific features?
- Board effects?
- Same power model?

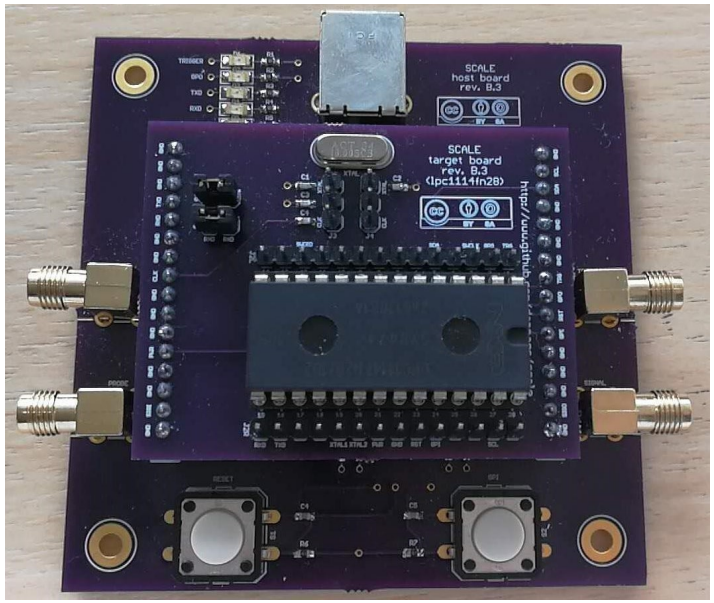
Power Model in LPC1114

SCALE: A customized side channel evaluation board for Cortex M0

- Target core: NXP LPC1114FN28
- 100 Ohm resistor on the VCC end
- On board amplifier/low pass filter
- Internal/external clock signal
- USB/external power supply

Power Model in LPC1114

Setup Comparison: STM32F0 v.s. NXP LPC1114



Power Model in LPC1114

Model building

- Same 23 thumb instructions as ELMO
- Testing mov-instr-mov (3-pipeline) 5000 times, with random operands
- Considering the operands as well as bus transition
- F-test determine whether a regression term is significant enough

Power Model in LPC1114

Regression Results

		adds	adds #imm	ands	cmp	cmp #imm	eors	ldr
	R^2	0.423	0.353	0.276	0.221	0.260	0.260	0.420
F-statistic	Operand 1	21.33	18.58	4.27	13.56	14.42	2.96	1.54
	Operand 2	54.30	0	37.95	9.83	0	22.44	104.26
	Transition 1	3.26	1.85	1.60	2.11	3.44	1.89	0.71
	Transition 2	32.78	61.14	12.88	16.70	35.07	24.75	1.15
	Overall	27.92	20.73	14.49	10.78	13.36	13.38	27.54
		ldrb	ldrh	lsls	lsrs	movs	movs #imm	muls
	R^2	0.422	0.345	0.240	0.284	0.335	0.196	0.628
F-statistic	Operand 1	0.94	1.01	35.91	43.27	1.63	17.94	44.43
	Operand 2	106.95	75.33	7.73	13.24	20.85	0	206.55
	Transition 1	0.77	1.47	1.27	1.11	2.51	1.28	2.04
	Transition 2	1.15	1.25	1.12	1.69	50.69	17.89	1.85
	Overall	27.81	20.07	12.03	15.07	19.19	9.29	64.25
		orrs	rors	str	strb	strh	subs	subs #imm
	R^2	0.416	0.385	0.158	0.493	0.230	0.331	0.207
F-statistic	Operand 1	12.63	51.09	16.64	137.03	40.40	7.31	9.84
	Operand 2	61.03	37.10	7.38	4.41	1.72	41.57	0
	Transition 1	3.75	2.24	1.26	1.08	1.13	1.94	3.28
	Transition 2	29.51	2.89	3.12	2.08	1.23	21.81	25.37
	Overall	27.13	23.79	7.12	36.95	11.37	18.83	9.96

Table 1: F-tests for significant joint data effects in the power consumption of the M0; tests which fail to reject at the 5% level are shaded grey

Power Model in LPC1114

Clustering power models for 23 instructions

STM32F0

- consistent with intuition
- ALU/Shift/Load/Store/Multiply

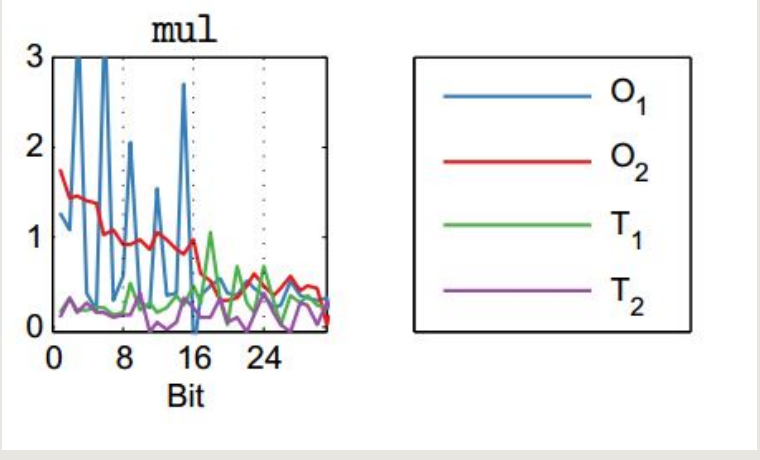
LPC1114

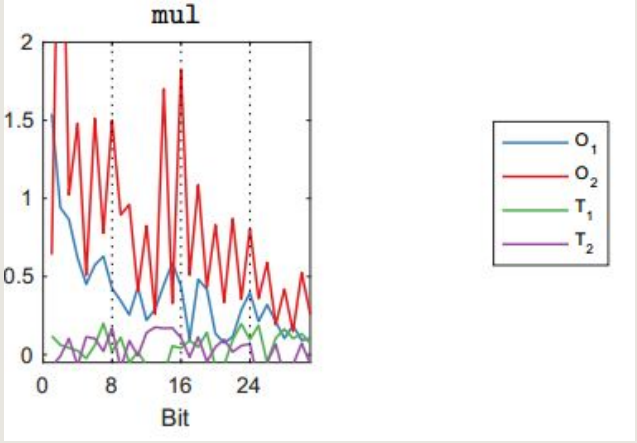
- most results consistent with intuition
- Shifts (LSL,LSR,LSLimm, etc.) have a separate cluster

Power Model in LPC1114

Bit-wise Comparison: most instructions are similar

MUL





Power Model in LPC1114

Conslusion

- Power model of LPC1114 is quite similar to STM32F0
- Security evaluation with ELMO is still reasonable
- Structural difference: Op1&Op2 in MUL
- Other individual features...

Timming Issues...

During the model building process, we constantly find that the leakage point varies among 23 instructions...

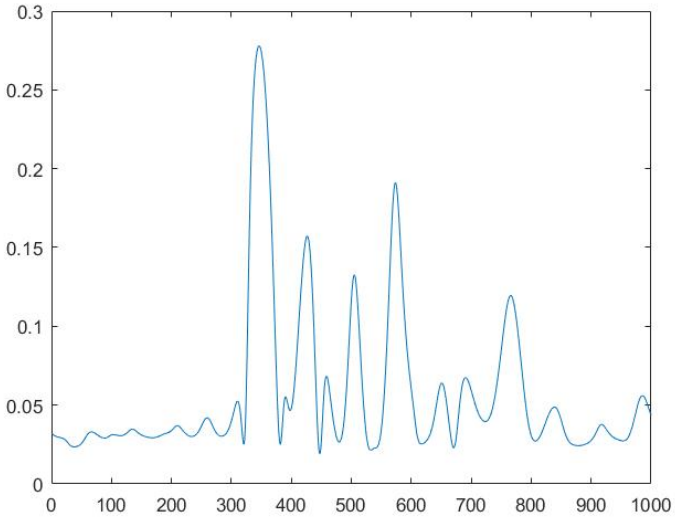


Figure: Leakage for *ands*

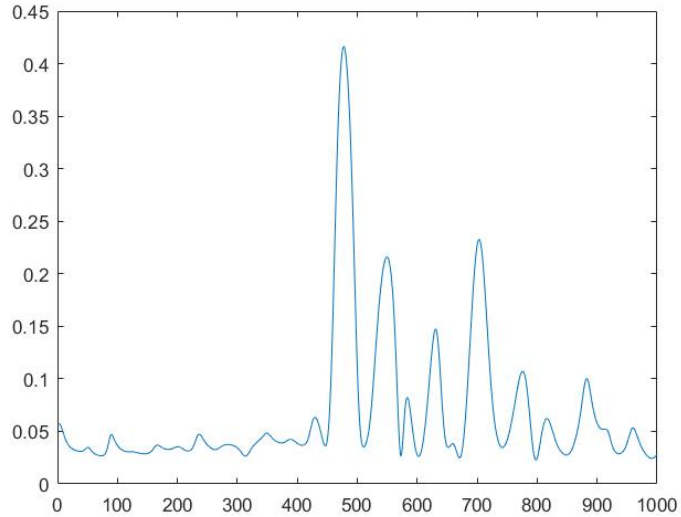


Figure: Leakage for *orrs*

Code layout & operating time

Same codes with different address alignment, learn the cycle count through SysTick:

5C	7C
1be: str r1, [r0, #0]	1c4: str r1, [r0, #0]
1c0: b.n 0x1c8	1c6: b.n 0x1ce
1c2: nop	1c8: nop
1c4: nop	1ca: nop
1c6: nop	1cc: nop
1c8: str r2, [r0, #0]	1ce: str r2, [r0, #0]

Reasoning: instruction pre-fetch

- Code stored in Flash
- Instruction-buffer:"...the Cortex-M0 has a smaller instruction buffer..."
- 0x1cc seems to be where the CPU finds the next fetching instruction is not in the 16B buffer
- Run from RAM: always constant time!

Consequence: branches

- Code layout affects the operation time
- Branches differ from not only their functionalities, but their layout
- Quite hard to patch; depends on the compiler

Consequence: Traceable implementation

Instruction pre-fetch leaves a peak on power trace:

Consequence: Traceable implementation

Instruction pre-fetch leaves a peak on power trace:

- Similar to trace-driven cache attack
- Acquisition with low resolution
- Leaves a road map of the CPU execution
- Useful tool for other attacks?

Future Work

- Power model for other series: Cortex M0+/M3
- Model quality: is hamming weight good enough?
- Board effect: will them affect security evaluation?

Acknowlegement

Horizon 2020
European Union funding
for Research & Innovation

References I

Onur Aciiçmez. “Yet another MicroArchitectural Attack: : exploiting I-Cache”. In: *Proceedings of the 2007 ACM workshop on Computer Security Architecture, CSAW 2007, Fairfax, VA, USA, November 2, 2007*. 2007, pp. 11–18.

Josep Balasch et al. “DPA, Bitslicing and Masking at 1 GHz”. In: *Cryptographic Hardware and Embedded Systems — CHES 2015: 17th International Workshop, Saint-Malo, France, September 13-16, 2015, Proceedings*. Ed. by Tim Güneysu and Helena Handschuh. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, pp. 599–619. ISBN: 978-3-662-48324-4.

References II

David Brumley and Dan Boneh. “Remote Timing Attacks Are Practical”. In: vol. 48. 5. New York, NY, USA: Elsevier North-Holland, Inc., 2005, pp. 701–716.

Daniel J. Bernstein. *Cache-timing attacks on AES*. 2005. URL: `http://cr.yp.to/antiforgery/cachetiming-20050414.pdf`.

Catherine H. Gebotys, Simon Ho, and C. C. Tiu. “EM Analysis of Rijndael and ECC on a Wireless Java-Based PDA”. In: *Cryptographic Hardware and Embedded Systems - CHES 2005, 7th International Workshop, Edinburgh, UK, August 29 - September 1, 2005, Proceedings*. 2005, pp. 250–264.

References III

Johann Großschädl et al. “Side-Channel Analysis of Cryptographic Software via Early-Terminating Multiplications”. In: *Information, Security and Cryptology - ICISC 2009, 12th International Conference, Seoul, Korea, December 2-4, 2009, Revised Selected Papers*. 2009, pp. 176–192.

Paul Kocher et al. “Spectre Attacks: Exploiting Speculative Execution”. In: *ArXiv e-prints* (Jan. 2018). arXiv:1801.01203.

References IV

Paul Kocher. "Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems". In: *Advances in Cryptology CRYPTO 96*. Ed. by Neal Koblitz. Vol. 1109. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, 1996. Chap. 9, pp. 104–113. ISBN: 978-3-540-61512-5.

Yuto Nakano et al. "A Pre-processing Composition for Secret Key Recovery on Android Smartphone". In: *Information Security Theory and Practice. Securing the Internet of Things - 8th IFIP WG 11.2 International Workshop, WISTP 2014, Heraklion, Crete, Greece, June 30 - July 2, 2014. Proceedings*. 2014, pp. 76–91.

References V

Dan Page. "Theoretical Use of Cache Memory as a Cryptanalytic Side-Channel." In: *IACR Cryptology ePrint Archive* 2002 (2002), p. 169. URL: http://dblp.uni-trier.de/db/journals/iacr/iacr2002.html#Page02.

Eran Tromer, Dag Arne Osvik, and Adi Shamir. "Efficient Cache Attacks on AES, and Countermeasures". In: *J. Cryptol.* 23.2 (2010), pp. 37–71. ISSN: 0933-2790.

Questions?