Energiegewahre Programmierung mit SEEP

Was wir erreicht haben

Laura Lawniczak, Johannes Schilling

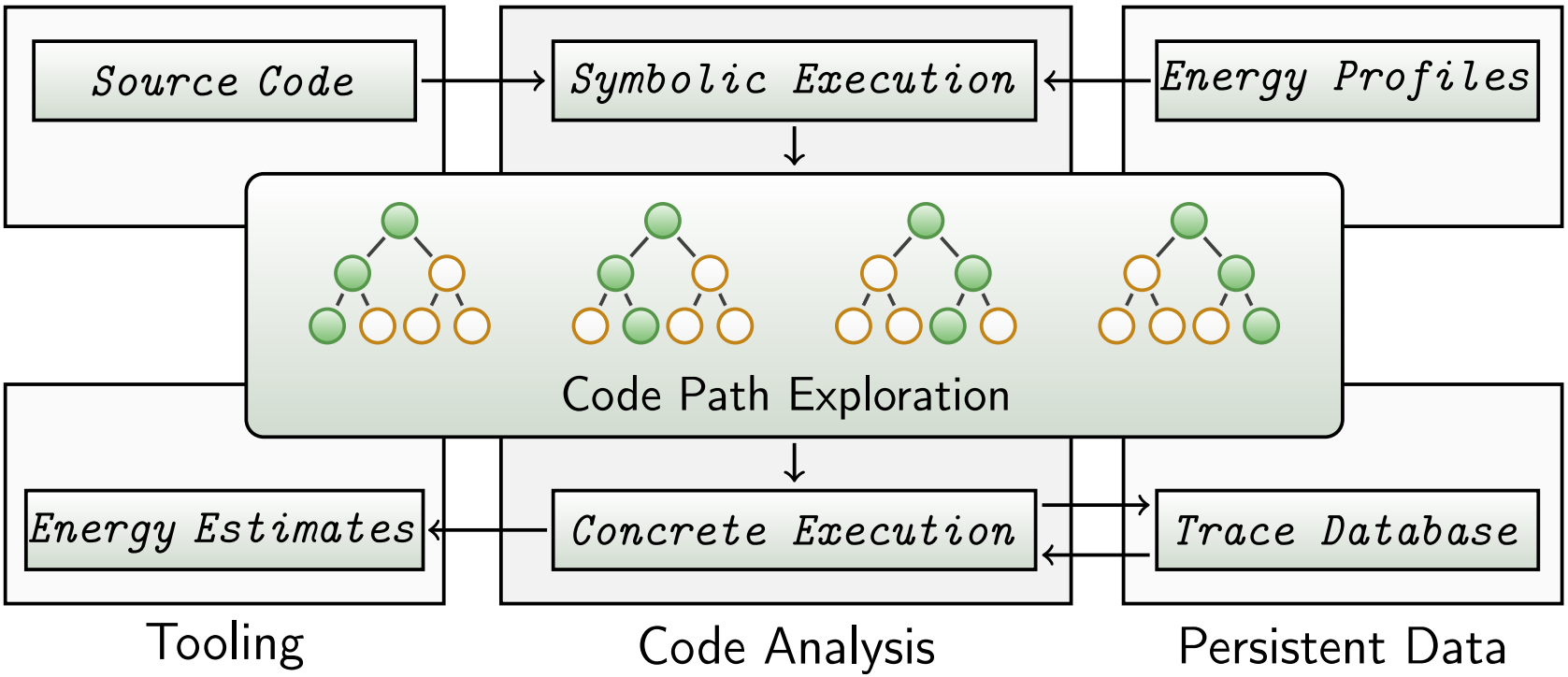
20. September 2013

Inhaltsverzeichnis

1. SEEP

1. SEEP

- Einsatz während der Softwareentwicklung
- Berechnet Energiewerte anhand plattformspezifischer Profile
- Pfadberechnung mithilfe von symbolischer Ausführung



2. Zeitplan

Ablaufplan

Tag 1, 2 Einarbeitung, Kandidaten zur Optimierung finden

- mit der Bedienung von SEEP vertraut machen
- mit SEEP das OS Contiki 2.6 evaluieren
- Funktionen zur Optimierung finden

Rest Woche 1 Optimierung der gefundenen Kandidaten

Woche 2 Messungen & Praktisches

- Ausführen des optimierten Codes auf dem MSP430 und Erstellung realer Energieprofile
- Vergleich der simulierten und gemessenen Energieprofile

Do & Fr Abschlussvortrag

Tatsächliche Version

Ablaufplan

Tag 1, 2	Einarbeitung
Rest Woche 1	Umplanen
Woche 2	Weitere Messungen, mehr AES
Fr	Abschlussvortrag

Ablaufplan

Tag 1, 2 **Einarbeitung**

- mit der Bedienung von SEEP vertraut gemacht
- SEEP-Unterstützung in Contiki OS 2.6
- Festgestellt, dass wir einen Großteil von Contiki nicht mit SEEP simulieren können

Rest Woche 1 Umplanen

Woche 2 Weitere Messungen, mehr AES

Fr Abschlussvortrag

Ablaufplan

Tag 1, 2	Einarbeitung
Rest Woche 1	**Umplanen**

- AES [1] als Contiki-App eingefügt
- SEEP-Fixes
- Freitag: Erstes Mal mit Messaufbau auseinander gesetzt

Woche 2	Weitere Messungen, mehr AES
Fr	Abschlussvortrag

Ablaufplan

Tag 1, 2 Einarbeitung

Rest Woche 1 Umplanen

Woche 2 **Weitere Messungen, mehr AES**

- Montag: andere AES-Implementierung [2] eingebaut
- Dienstag: alle noch fehlenden Einzelinstruktionen für das Energieprofil nachgemessen
- Mittwoch: AES-Implementierungen miteinander verglichen
- Donnerstag: SEEP-Ergebnisse mit Realmessungen verglichen

Fr Abschlussvortrag

Ablaufplan

Tag 1, 2	Einarbeitung
Rest Woche 1	Umplanen
Woche 2	Weitere Messungen, mehr AES
Fr	**Abschlussvortrag**

- Nachbereitung der gewonnenen Ergebnisse und Erfahrungen
- Anfertigen des Abschlussvortrags
- Abschlussvortrag halten

3. Contiki 2.6 mit SEEP

3. Contiki 2.6 mit SEEP

- Mehrdimensionale Arrays und Structs werden von SEEP nicht unterstützt
- Nicht unterstützte Methoden müssen ignoriert werden oder erfordern manuelle Instrumentierung
- Contiki verwendet viele Hardwarezugriffe, Netzwerk etc.
- Probleme bei der symbolischen Ausführung
- In SEEP schlecht/nicht analysierbar

Ausnahmeliste

```
> grep -v '^$' exceptionList | grep -v '^#' | wc -l
42
```

Evaluierte Methoden (beispielhaft)

```
--> Printing statistics:
    +++ Number of invocations per function:
        o puts: 5
        o setvbuf: 1
        o autostart_start: 1
        o ctimer_init: 1
        o procinit_init: 1
        o serial_line_init: 1
        o ringbuf_init: 1
        o netstack_init: 1
        o process_start: 1
        o process_init: 1
```

Ausgaben von SEEP

Scorings

```
MixSubColumns()
-- Number of invocations: 9
   0.PS    PeakScorer        4451     Amount of energy consumed: 40059 nJ
   1.MS    MedianScorer      4451     Amount of energy consumed: 40059 nJ
```

Konkrete Energiewerte

```
* Estimated overall energy value for function
            ./hello-worldMSP430ContikiOOEncrypt___0_31920216_: 52736 nJ
```

Assemblerinstruktionen

```
321x movrr: 1.28368 nJ * 321 = 412.061 nJ
1234x movridx: 3.28228 nJ * 1234 = 4050.33 nJ
2958x movrir: 3.28228 nJ * 2958 = 9708.98 nJ
2467x movria: 3.60359 nJ * 2467 = 8890.06 nJ
```

4. AES

4. AES

- 128 Bit Schlüssellänge, da von TelosB Mote unterstützt
- 2 Software-Implementierungen [1, 2]

4. AES

- 128 Bit Schlüssellänge, da von TelosB Mote unterstützt
- 2 Software-Implementierungen [1, 2]

Algorithmus

1. SubBytes – Byte-weise durch Werte aus der S-Box ersetzen
2. ShiftRows – Zeilenverschiebung im aktuellen Block
3. MixColumns – Innerhalb der Spalte Modulo-Multiplikation, XOR
4. AddRoundKey – XOR mit aktuellem Rundenschlüssel

das Ganze n mal

Verschiedene Implementierungen

Hardware-Implementierung

- TelosB Mote hat ein CC2420-Modul, das 128-Bit-AES-CTR unterstützt (IEEE 802.15.4)
- logischerweise nicht mit SEEP evaluierbar

Software-Implementierung [1]

- verwendet `structs`, mehrdimensionale Arrays
- nur begrenzt mit SEEP evaluierbar

Software-Implementierung [2]

- alles^{TM} `uchar`-Arrays

Verschiedene Implementierungen

Hardware-Implementierung

- TelosB Mote hat ein CC2420-Modul, das 128-Bit-AES-CTR unterstützt (IEEE 802.15.4)
- logischerweise nicht mit SEEP evaluierbar

Software-Implementierung [1]

- verwendet `structs`, mehrdimensionale Arrays
- nur begrenzt mit SEEP evaluierbar

Software-Implementierung [2]

- alles^{TM} `uchar`-Arrays
- relativ gut mit SEEP evaluierbar

Verschiedene Implementierungen

Hardware-Implementierung

- TelosB Mote hat ein CC2420-Modul, das 128-Bit-AES-CTR unterstützt (IEEE 802.15.4)
- logischerweise nicht mit SEEP evaluierbar

Software-Implementierung [1]

- verwendet `structs`, mehrdimensionale Arrays
- nur begrenzt mit SEEP evaluierbar

Software-Implementierung [2]

- alles^{TM} `uchar`-Arrays
- relativ gut mit SEEP evaluierbar
- relativ optimal was Energieverbrauch angeht

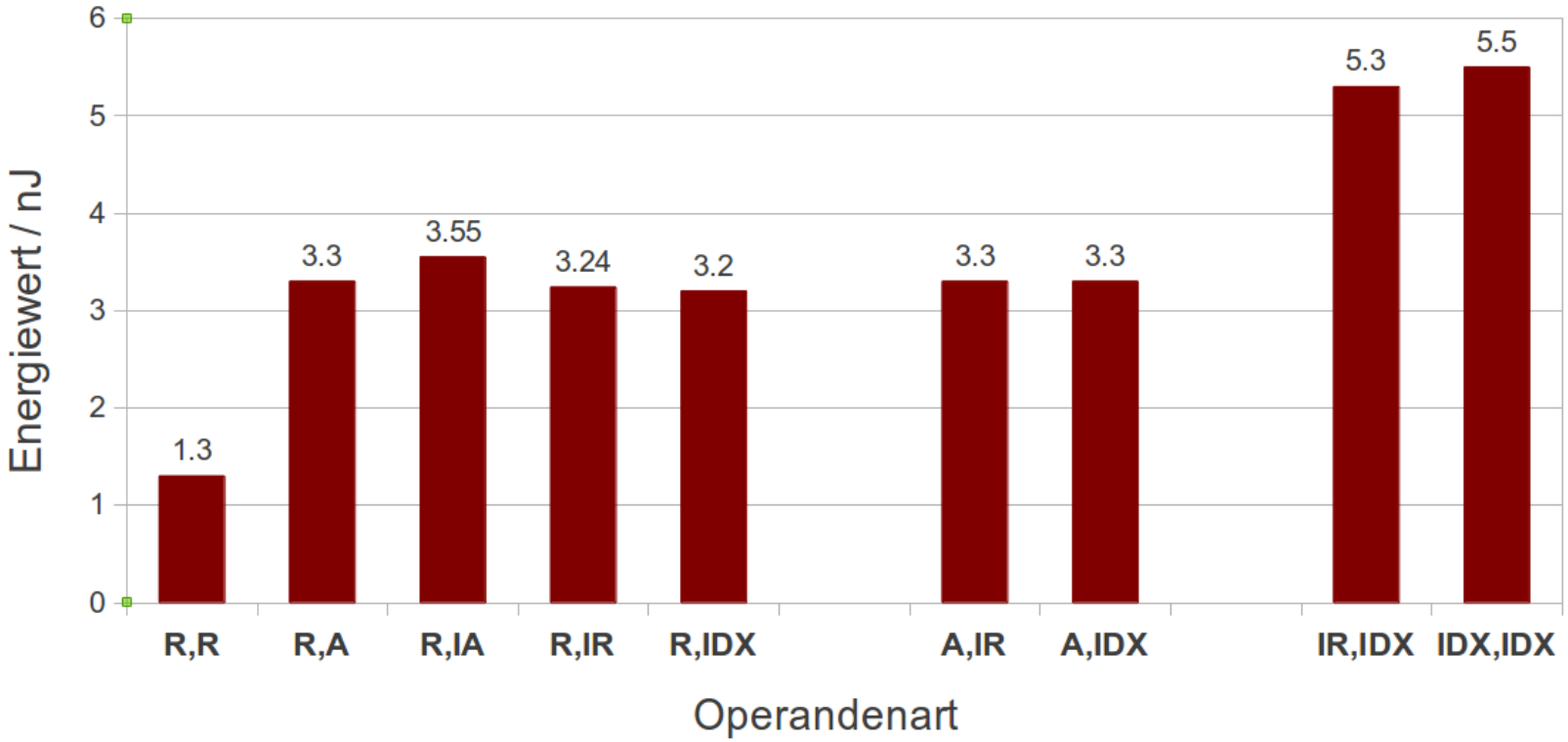
5. Messungen

- Erweitern des Profils für den MSP430
- Eine Messung muss signifikant länger als 150 ms dauern
- Oft mehr als 65535 ($2^{16} - 1$) Instruktionen nötig

Messen von einzelnen Assemblerinstruktionen

```
asm volatile (
    "mov     #65535, r5"
    "loop0:"
    "add     #2, r6"
    "add     #2, r6"
    ...
    "sub     #1, r5"
    "cmp     #0, r5"
    "jlt     loop0"
);
```

R=Register; **A**=Absolute Value; **IA**=Immediate Address;
IR=Indirekt Register; **IDX**=Indexed



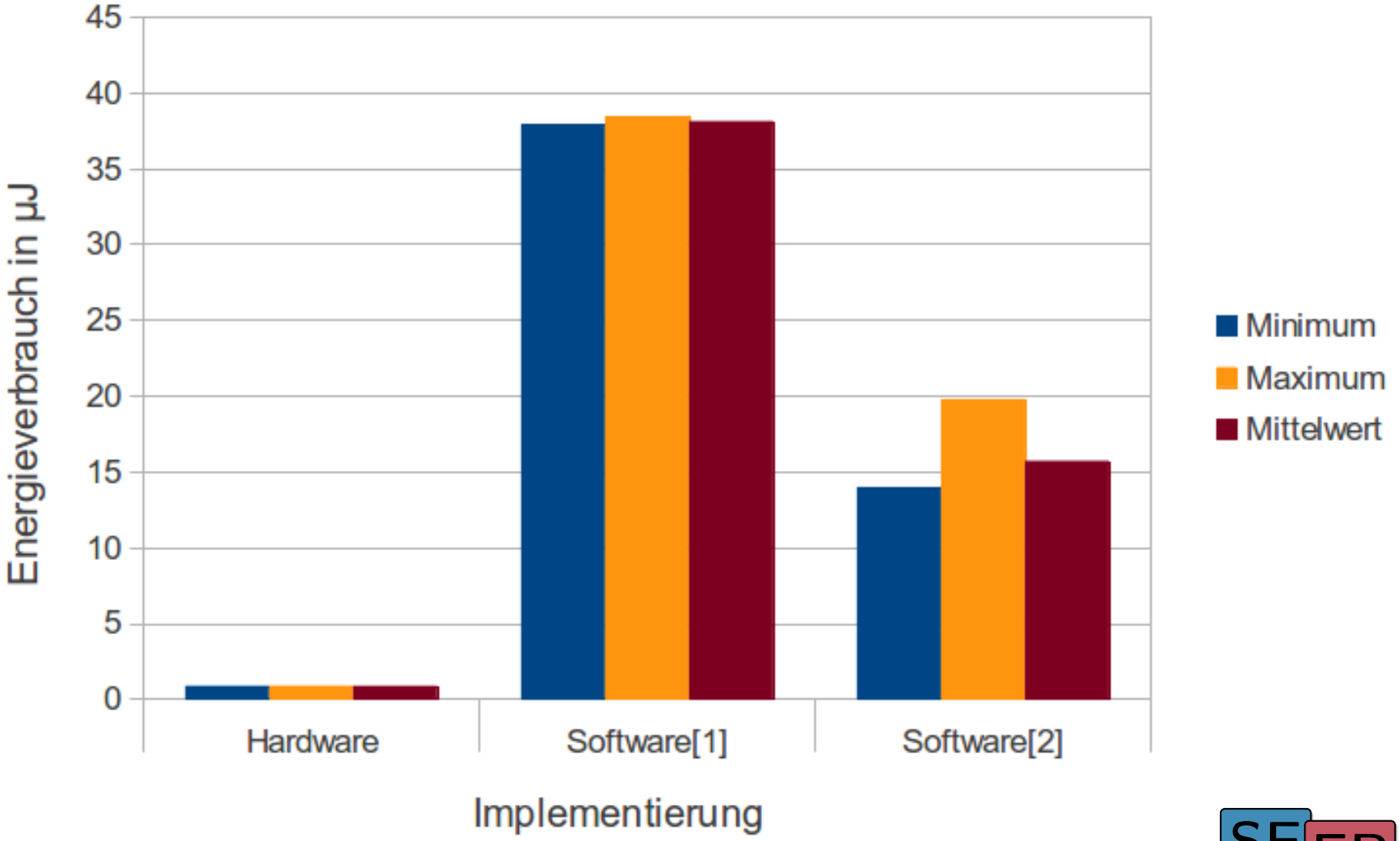
AES – Ergebnisse

- Messungen mit demselben Schlüssel / derselben Eingabe
- Schleifendurchläufe:
  - Hardware: 6553500
  - Software: 65535

Ergebnisse im Vergleich

Implementierung	Energieverbrauch in μJ		
	Minimum	Maximum	Mittelwert
Hardware	0.857023	0.862233	0.858993
Software [1]	37.8724	38.3987	38.1053
Software [2]	13.9699	19.7641	15.6922

AES – Ergebnisse



6. Vergleich: SEEP vs. Messungen

SEEP mit -O0

```
MixSubColumns()
-- Number of invocations: 9
   0.PS    PeakScorer     4451     Amount of energy consumed: 40059 nJ
   1.MS    MedianScorer   4451     Amount of energy consumed: 40059 nJ
Encrypt()
-- Number of invocations: 1
   0.PS    PeakScorer     52736     Amount of energy consumed: 52736 nJ
   1.MS    MedianScorer   52736     Amount of energy consumed: 52736 nJ
```

SEEP mit -O0

```
MixSubColumns()
-- Number of invocations: 9
   0.PS    PeakScorer     4451     Amount of energy consumed: 40059 nJ
   1.MS    MedianScorer   4451     Amount of energy consumed: 40059 nJ
Encrypt()
-- Number of invocations: 1
   0.PS    PeakScorer     52736     Amount of energy consumed: 52736 nJ
   1.MS    MedianScorer   52736     Amount of energy consumed: 52736 nJ
```

SEEP mit -Os

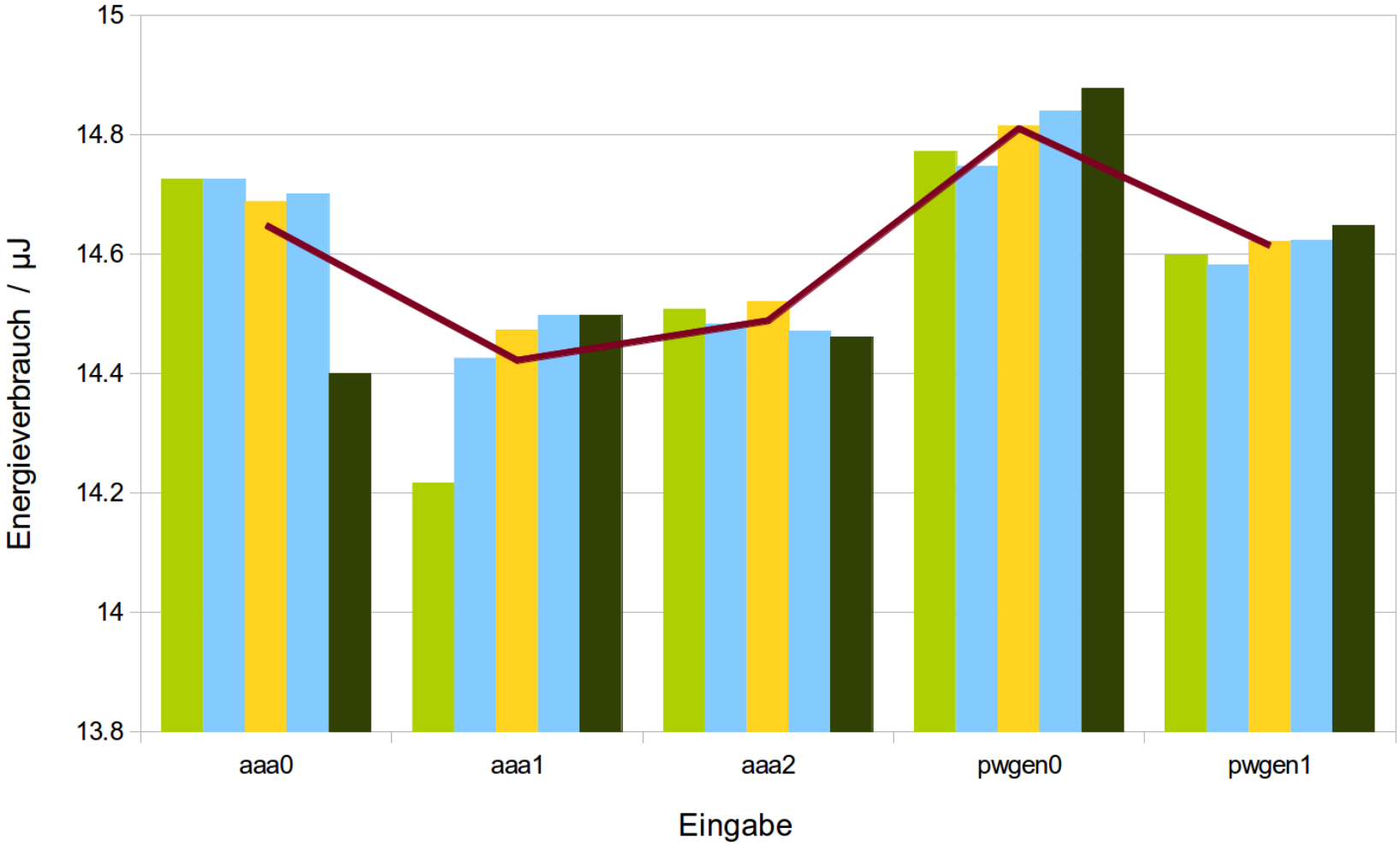
```
MixSubColumns()
-- Number of invocations: 9
   0.PS    PeakScorer     4647     Amount of energy consumed: 41823 nJ
   1.MS    MedianScorer   4647     Amount of energy consumed: 41823 nJ
Encrypt()
-- Number of invocations: 1
   0.PS    PeakScorer     16216     Amount of energy consumed: 16216 nJ
   1.MS    MedianScorer   16216     Amount of energy consumed: 16216 nJ
```

7. Ausblick, Fazit

Seitenkanal-Angriffe

- Verschiedene Ausführungszeiten für verschiedene Eingaben
- Rückschlüsse aus der Speicher-/Cache-Nutzung auf Eingabe
- NEU: Rückschlüsse aus Energieverbrauch auf Eingabe

7. Ausblick, Fazit



Was man noch machen wollen könnte

Seitenkanal-Angriffe

- Verschiedene Ausführungszeiten für verschiedene Eingaben
- Rückschlüsse aus der Speicher-/Cache-Nutzung auf Eingabe
- NEU: Rückschlüsse aus Energieverbrauch auf Eingabe

Nachgemessen

- Energieverbrauch bei Zufallseingabe gefühlt höher
- Varianz recht groß, v.a. auch innerhalb der aaaa...-Durchläufe

mov is Turing-Complete!

Stephen Dolan, Computer Laboratory, University of Cambridge

mov is Turing-Complete!

Stephen Dolan, Computer Laboratory, University of Cambridge

Alles einfacher

- n^2 Energiewerte für n Operandentypen
- Profile einfacher, schneller und effizienter erstellen
- Profile brauchen weniger Speicherplatz
- Keine Backups mehr notwendig, Profile werden auswendig gelernt

Vielen Dank für die Aufmerksamkeit

Fragen?

Quellen

FIPS-197 compliant AES implementation, Copyright (C) 2006-2013, Brainspark B.V., GPLv2+ `https://polarssl.org/aes-source-code`, abgerufen am 19.9.2013

A public domain byte-oriented implementation of AES in C, karl malbrain, malbrain@yahoo.com, public domain `https://code.google.com/p/byte-oriented-aes/downloads/detail?name=aestable.c`, abgerufen am 19.9.2013