RUHR UNIVERSITÄT BOCHUM RUB

hgi SysSec

Everything you wanted to know about x86 microcode - but might have been afraid to ask

34th Chaos Communication Congress, Leipzig

December 28, 2017

Philipp Koppe, Benjamin Kollenda, Marc Fyrbiak, Christian Kison,
Robert Gawlik, Christof Paar, Thorsten Holz

Horst Görtz Institute for IT-Security
Ruhr-Universität Bochum
<firstname.lastname>@rub.de

emproof
www.emproof.de

Outline

RUHR UNIVERSITÄT BOCHUM RUB

- What is microcode?
- Architectural crash course
- Is it hackable?
- Demo

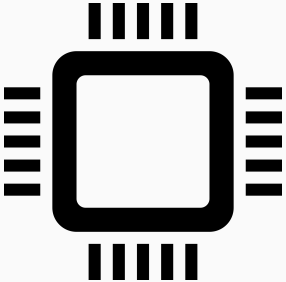
Outline

RUHR UNIVERSITÄT BOCHUM

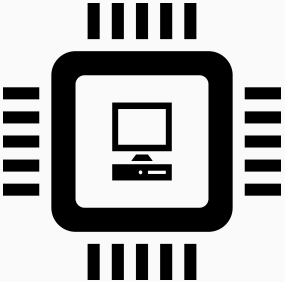
RUB

- What is microcode?
- Architectural crash course
- Is it hackable?
- Demo

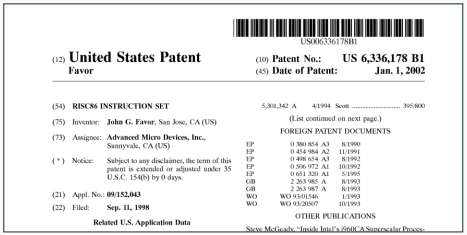
What is microcode?



What is microcode?



Previous Work



US006336178B1

(12) **United States Patent**
Favor

(10) **Patent No.: US 6,336,178 B1**
(45) **Date of Patent: Jan. 1, 2002**

(54) **RISC86 INSTRUCTION SET**

(75) Inventor: **John G. Favor**, San Jose, CA (US)

(73) Assignee: **Advanced Micro Devices, Inc.**, Sunnyvale, CA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

(21) Appl. No.: **09/152,043**

(22) Filed: **Sep. 11, 1998**

Related U.S. Application Data

5,301,342 A 4/1994 Scott 395/800

(List continued on next page.)

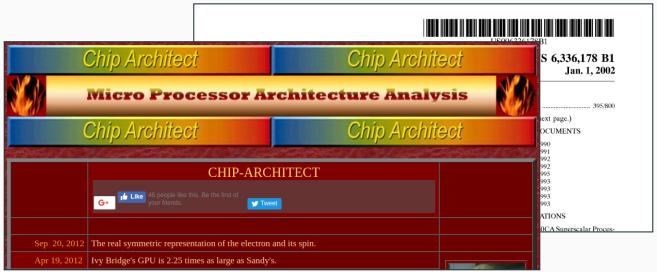
FOREIGN PATENT DOCUMENTS

EP	0 380 854 A3	8/1990
EP	0 454 984 A2	11/1991
EP	0 498 654 A3	8/1992
EP	0 506 972 A1	10/1992
EP	0 651 320 A1	5/1995
GB	2 263 985 A	8/1993
GB	2 263 987 A	8/1993
WO	WO 93/01546	1/1993
WO	WO 93/20507	10/1993

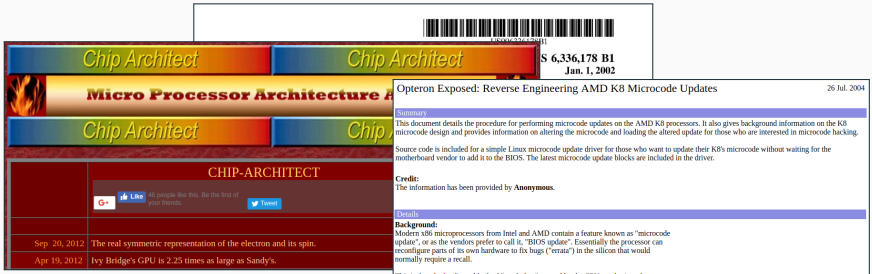
OTHER PUBLICATIONS

Steve McGeady, "Inside Intel's 486DA Superscalar Proces-

Previous Work



Previous Work



US 6,336,178 B1
Jan. 1, 2002

Chip Architect

Micro Processor Architecture

CHIP-ARCHITECT

Like 46 people like this. Be the first of your friends.

Tweet

Sep 20, 2012	The real symmetric representation of the electron and its spin.
Apr 19, 2012	Ivy Bridge's GPU is 2.25 times as large as Sandy's.

Opteron Exposed: Reverse Engineering AMD K8 Microcode Updates

26 Jul. 2004

Summary

This document details the procedure for performing microcode updates on the AMD K8 processors. It also gives background information on the K8 microcode design and provides information on altering the microcode and loading the altered update for those who are interested in microcode hacking.

Source code is included for a simple Linux microcode update driver for those who want to update their K8's microcode without waiting for the motherboard vendor to add it to the BIOS. The latest microcode update blocks are included in the driver.

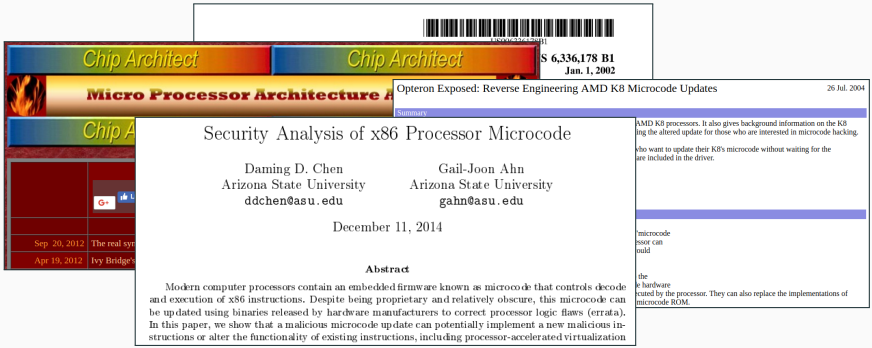
Credit:
The information has been provided by **Anonymous**.

Details

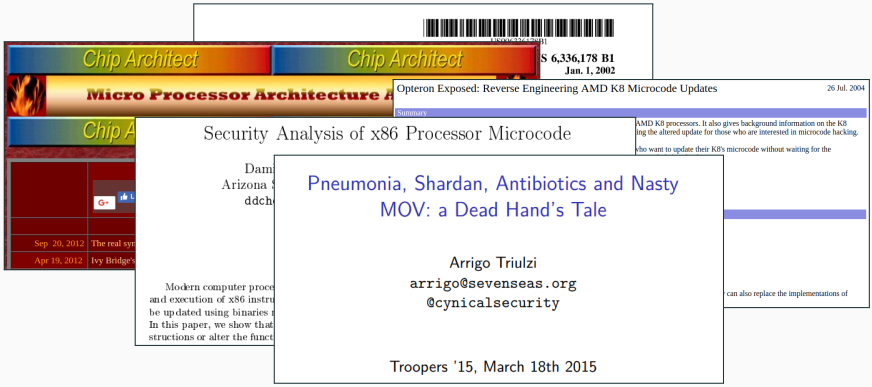
Background:
Modern x86 microprocessors from Intel and AMD contain a feature known as "microcode update", or as the vendors prefer to call it, "BIOS update". Essentially the processor can reconfigure parts of its own hardware to fix bugs ("errata") in the silicon that would normally require a recall.

This is done by loading a block of "patch data" created by the CPU vendor into the processor using special control registers. Microcode updates essentially override hardware features with sequences of the internal RISC-like micro-ops (uops) actually executed by the processor. They can also replace the implementations of microcoded instructions already handled by hard-wired sequences in an on-die microcode ROM.

Previous Work



Previous Work



What is it used for?

What is it used for?

- Instruction decoding

What is it used for?

- Instruction decoding
- Fix CPU bugs

What is it used for?

- Instruction decoding
- Fix CPU bugs
- Exception handling

What is it used for?

- Instruction decoding
- Fix CPU bugs
- Exception handling
- Power Management

What is it used for?

- Instruction decoding
- Fix CPU bugs
- Exception handling
- Power Management
- Complex features (Intel SGX)

x86 ISA is complex

RUHR UNIVERSITÄT BOCHUM

RUB

```
C3                              ret
```

x86 ISA is complex

RUHR UNIVERSITÄT BOCHUM

RUB

```
C3                          ret

48 b8 88 77 66 55           movabs   rax,0x1122334455667788
44 33 22 11
```

x86 ISA is complex

RUHR UNIVERSITÄT BOCHUM

RUB

```
C3                          ret

48 b8 88 77 66 55           movabs   rax,0x1122334455667788
44 33 22 11

64 ff 03                    inc DWORD PTR fs:[ebx]
```

x86 ISA is complex

RUHR UNIVERSITÄT BOCHUM

RUB

```
C3                          ret

48 b8 88 77 66 55           movabs   rax,0x1122334455667788
44 33 22 11

64 ff 03                    inc DWORD PTR fs:[ebx]

64 67 66 f0 ff 07           lock inc WORD PTR fs:[bx]
```

x86 ISA is complex

RUHR UNIVERSITÄT BOCHUM RUB

```
C3                          ret

48 b8 88 77 66 55           movabs   rax,0x1122334455667788
44 33 22 11

64 ff 03                    inc DWORD PTR fs:[ebx]

64 67 66 f0 ff 07           lock inc WORD PTR fs:[bx]

2e c4 e2 71 96 84           vfmaddsub132ps xmm0, xmm1,
be 34 23 12 01              xmmword ptr cs:
                            [esi + edi * 4 + 0x11223344]
```

Micro Ops

RUHR UNIVERSITÄT BOCHUM

RUB

```
pop [ebx]
```

Micro Ops

RUHR UNIVERSITÄT BOCHUM

RUB

```
pop [ebx]
```

```
load temp, [esp]
store [ebx], temp
add esp, 4
```

x86 CPUs are prone to errors

x86 CPUs are prone to errors

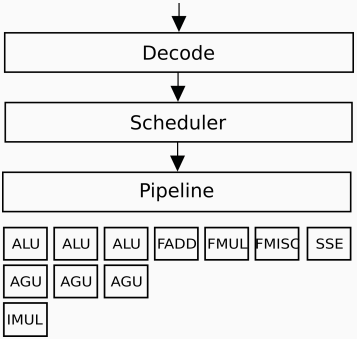
x86 CPUs are prone to errors

x86 CPUs are prone to errors

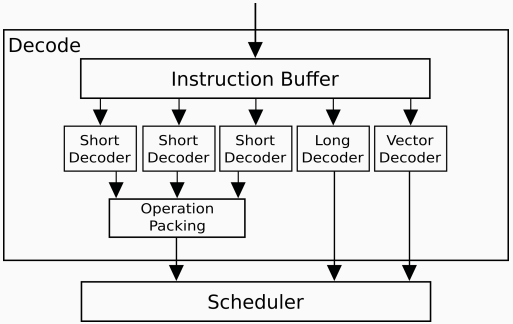
Outline

- What is microcode?
- Architectural crash course
- Is it hackable?
- Demo

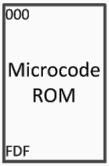
x86 Instruction Decoding



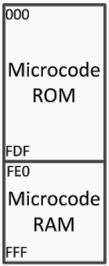
x86 Instruction Decoding



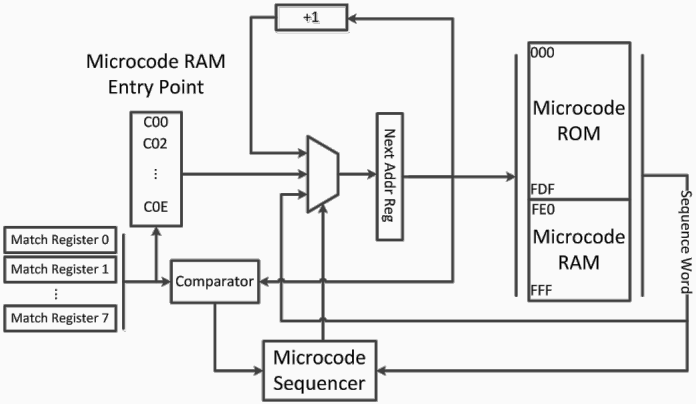
Microcode Engine (Vector Decoder)



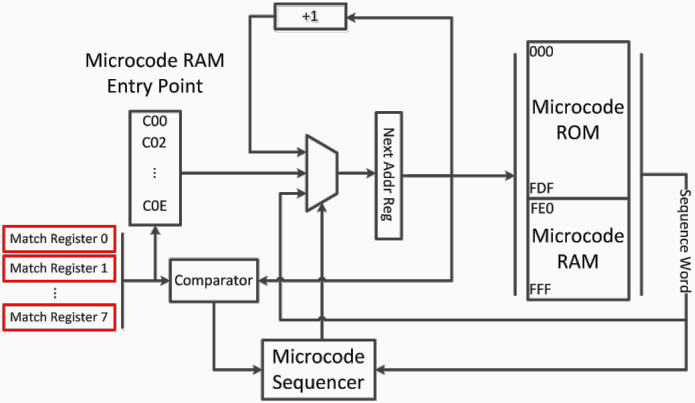
Microcode Engine (Vector Decoder)



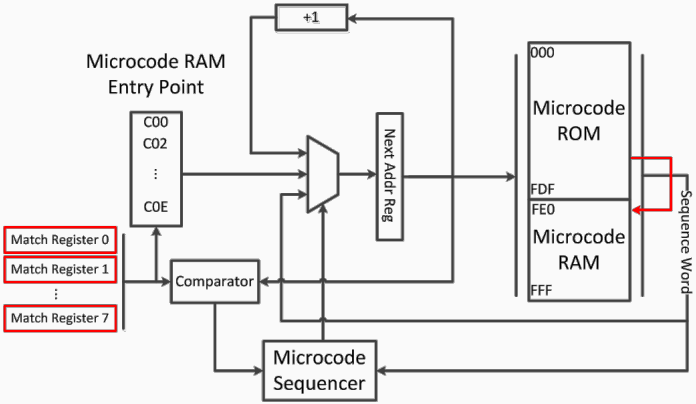
Microcode Engine (Vector Decoder)



Microcode Engine (Vector Decoder)



Microcode Engine (Vector Decoder)



Microcode Update Mechanism

RUHR UNIVERSITÄT BOCHUM RUB

- Kernel mode
- Load microcode update into RAM
- Write virtual address to MSR `0xC0010020`
- Microcode patches not persistent

Microcode Update File Format

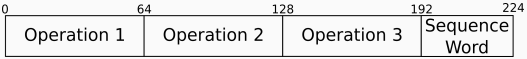
RUHR UNIVERSITÄT BOCHUM RUB

B↓ Bit→	0 31	32 63
0	date	patch ID
8	patch block | len | init	checksum
16	northbridge ID	southbridge ID
24	CPUID	magic value
32	match register 0	match register 1
40	match register 2	match register 3
48	match register 4	match register 5
54	match register 6	match register 7
64	triad 0, microinstruction 0	
72	triad 0, microinstruction 1	
80	triad 0, microinstruction 2	
88	triad 0, sequence word	triad 1 ...

Microcode Update File Format

RUHR UNIVERSITÄT BOCHUM RUB

B↓ Bit→	0 31	32 63
0	date	patch ID
8	patch block | len | init	checksum
16	northbridge ID	southbridge ID
24	CPUID	magic value
32	match register 0	match register 1
40	match register 2	match register 3
48	match register 4	match register 5
54	match register 6	match register 7
64	triad 0, microinstruction 0	
72	triad 0, microinstruction 1	
80	triad 0, microinstruction 2	
88	triad 0, sequence word	triad 1 ...



Outline

RUHR UNIVERSITÄT BOCHUM

RUB

- What is microcode?
- Architectural crash course
- Is it hackable?
- Demo

Is it hackable?

Is it hackable?

RUHR UNIVERSITÄT BOCHUM RUB

- CPUs updatable

Is it hackable?

- CPUs updatable
- Update drivers in Linux kernel

Is it hackable?

- CPUs updatable
- Update drivers in Linux kernel
- Microcode updates

Is it hackable?

- CPUs updatable
- Update drivers in Linux kernel
- Microcode updates
- Update file format

Is it hackable?

- CPUs updatable
- Update drivers in Linux kernel
- Microcode updates
- Update file format
- Hints that there is no strong crypto

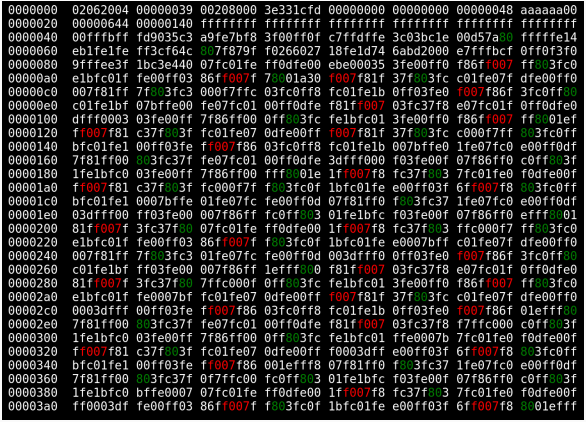
Unknown Instruction Set

RUHR UNIVERSITÄT BOCHUM RUB

```
0000000  02062004 00000039 00208000 3e331cfd 00000000 00000000 00000048 aaaaaa00
0000020  00000644 00000140 ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff
0000040  00fffbff fd9035c3 a9fe7bf8 3f00ff0f c7ffdffe 3c03bc1e 00d57a80 ffffe14
0000060  eb1fe1fe ff3cf64c 807f879f f0266027 18fe1d74 6abd2000 e7fffbcf 0ff0f3f0
0000080  9fffee3f 1bc3e440 07fc01fe ff0dfe00 ebe00035 3fe00ff0 f86ff007 ff803fc0
00000a0  e1bfc01f fe00ff03 86ff007f 78001a30 f007f81f 37f803fc c01fe07f dfe00ff0
00000c0  007f81ff 7f803fc3 000f7ffc 03fc0ff8 fc01fe1b 0ff03fe0 f007f86f 3fc0ff80
00000e0  c01fe1bf 07bffe00 fe07fc01 00ff0dfe f81ff007 03fc37f8 e07fc01f 0ff0dfe0
0000100  dfff0003 03fe00ff 7f86ff00 0ff803fc fe1bfc01 3fe00ff0 f86ff007 ff8001ef
0000120  ff007f81 c37f803f fc01fe07 0dfe00ff f007f81f 37f803fc c000f7ff 803fc0ff
0000140  bfc01fe1 00ff03fe ff007f86 03fc0ff8 fc01fe1b 007bffe0 1fe07fc0 e00ff0df
0000160  7f81ff00 803fc37f fe07fc01 00ff0dfe 3dfff000 f03fe00f 07f86ff0 c0ff803f
0000180  1fe1bfc0 03fe00ff 7f86ff00 fff8001e 1ff007f8 fc37f803 7fc01fe0 f0dfe00f
00001a0  ff007f81 c37f803f fc000f7f f803fc0f 1bfc01fe e00ff03f 6ff007f8 803fc0ff
00001c0  bfc01fe1 0007bffe 01fe07fc fe00ff0d 07f81ff0 f803fc37 1fe07fc0 e00ff0df
00001e0  03dfff00 ff03fe00 007f86ff fc0ff803 01fe1bfc f03fe00f 07f86ff0 efff8001
0000200  81ff007f 3fc37f80 07fc01fe ff0dfe00 1ff007f8 fc37f803 ffc000f7 ff803fc0
0000220  e1bfc01f fe00ff03 86ff007f f803fc0f 1bfc01fe e0007bff c01fe07f dfe00ff0
0000240  007f81ff 7f803fc3 01fe07fc fe00ff0d 003dfff0 0ff03fe0 f007f86f 3fc0ff80
0000260  c01fe1bf ff03fe00 007f86ff 1efff800 f81ff007 03fc37f8 e07fc01f 0ff0dfe0
0000280  81ff007f 3fc37f80 7ffc000f 0ff803fc fe1bfc01 3fe00ff0 f86ff007 ff803fc0
00002a0  e1bfc01f fe0007bf fc01fe07 0dfe00ff f007f81f 37f803fc c01fe07f dfe00ff0
00002c0  0003dfff 00ff03fe ff007f86 03fc0ff8 fc01fe1b 0ff03fe0 f007f86f 01efff80
00002e0  7f81ff00 803fc37f fe07fc01 00ff0dfe f81ff007 03fc37f8 f7ffc000 c0ff803f
0000300  1fe1bfc0 03fe00ff 7f86ff00 0ff803fc fe1bfc01 ffe0007b 7fc01fe0 f0dfe00f
0000320  ff007f81 c37f803f fc01fe07 0dfe00ff f0003dff e00ff03f 6ff007f8 803fc0ff
0000340  bfc01fe1 00ff03fe ff007f86 001efff8 07f81ff0 f803fc37 1fe07fc0 e00ff0df
0000360  7f81ff00 803fc37f 0f7ffc00 fc0ff803 01fe1bfc f03fe00f 07f86ff0 c0ff803f
0000380  1fe1bfc0 bffe0007 07fc01fe ff0dfe00 1ff007f8 fc37f803 7fc01fe0 f0dfe00f
00003a0  ff0003df fe00ff03 86ff007f f803fc0f 1bfc01fe e00ff03f 6ff007f8 8001efff
```

Unknown Instruction Set

Unknown Instruction Set



Is it hackable?

- Update drivers in Linux kernel
- Microcode updates
- Update file format
- Hints that there is no strong crypto

Is it hackable?

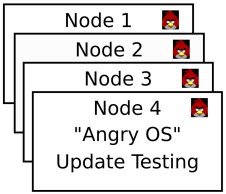
- Update drivers in Linux kernel
- Microcode updates
- Update file format
- Hints that there is no strong crypto
- CPU accepts modified updates

Is it hackable?

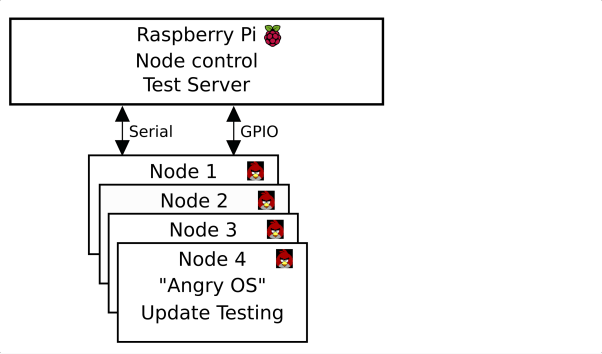
RUHR UNIVERSITÄT BOCHUM RUB

- Update drivers in Linux kernel
- Microcode updates
- Update file format
- Hints that there is no strong crypto
- CPU accepts modified updates
- **Yes!**

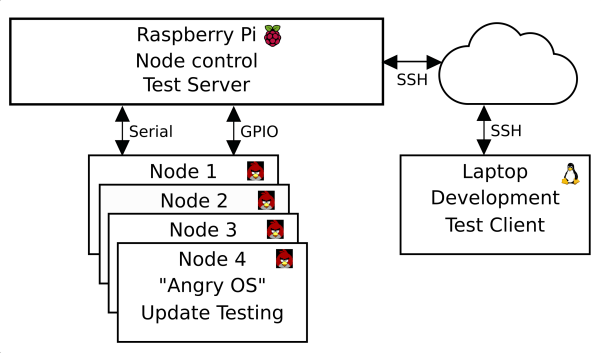
Framework

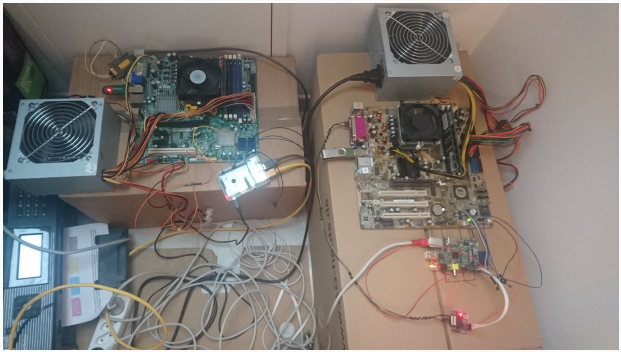


Framework

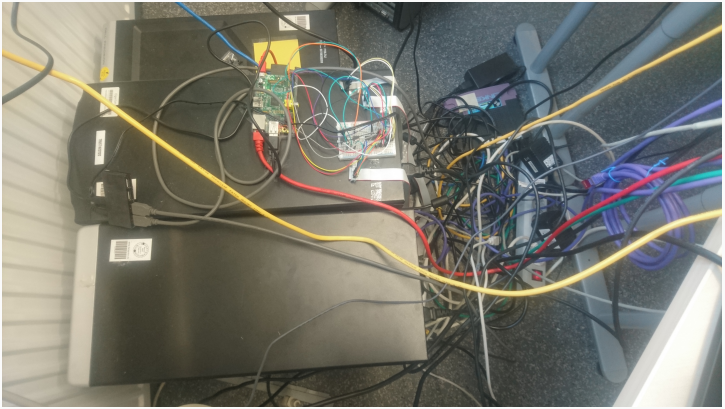

Raspberry Pi
Node control
Test Server
Serial
GPIO
Node 1
Node 2
Node 3
Node 4
"Angry OS"
Update Testing

Framework


Raspberry Pi
Node control
Test Server
SSH
Serial
GPIO
SSH
Node 1
Node 2
Node 3
Node 4
"Angry OS"
Update Testing
Laptop
Development
Test Client



Framework



Heatmaps

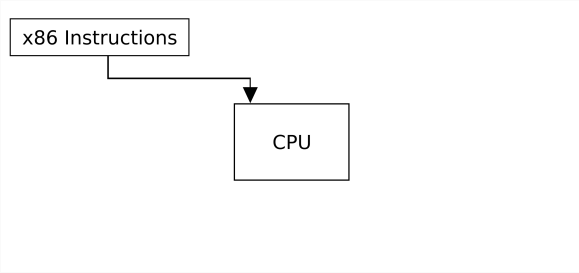
RUHR UNIVERSITÄT BOCHUM RUB

ROM Address	vector instruction
0x900 - 0x913	-
0x900 - 0x913	-
0x914 - 0x917	rep_cmps_mem8
0x918 - 0x95f	-
0x960	mul_mem16
0x961	idiv
0x962	mul_reg16
0x963	-
0x964	imul_mem16
0x965	bound
0x966	imul_reg16
0x967	-
0x968	bts_imm
0x969 - 0x971	-
0x972 - 0x973	div
0x974 - 0x975	-
0x976 - 0x977	idiv
0x978	-
0x979 - 0x97a	idiv
0x97b - 0x9a7	-
0x9a8	btr_imm
0x9a9 - 0x9ad	-
0x9ae	mfence
0x9af - 09ff	-

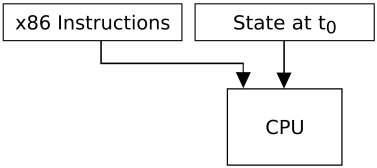
Reverse Engineering Setting

- Unknown instruction set analysis
- Black box model with oracle
- Feed inputs, filter and observe outputs
- Infer structure, encoding, meaning

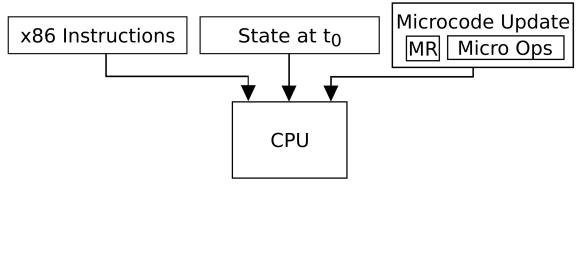
Processor Oracle



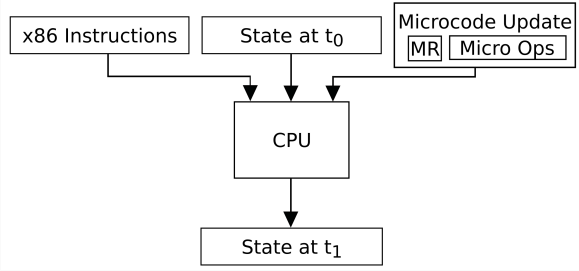
Processor Oracle



Processor Oracle



Processor Oracle



Early attempts

RUHR UNIVERSITÄT BOCHUM RUB

Input:

```
eax 00000101 ebx 00000101 ecx 00000102 edx 00000103
esi 00000104 edi 00000105 ebp 00000106 esp 0013b4??
```

```
00000000000001111101000000011110110000000000000000000000011010101
```

Early attempts

RUHR UNIVERSITÄT BOCHUM RUB

Input:

```
eax 00000101 ebx 00000101 ecx 00000102 edx 00000103
esi 00000104 edi 00000105 ebp 00000106 esp 0013b4??
```

```
00000000000001111101000000011110110000000000000000000011010101
```

Output:

```
eax 000001d6 ebx 00000101 ecx 00000102 edx 00000103
esi 00000104 edi 00000105 ebp 00000106 esp 0013b4??
```

Early attempts

RUHR UNIVERSITÄT BOCHUM RUB

Input:

```
eax 00000101 ebx 00000101 ecx 00000102 edx 00000103
esi 00000104 edi 00000105 ebp 00000106 esp 0013b4??
```

```
00000000000001111101000000011110110000000000000000000000001010101
```

Early attempts

RUHR UNIVERSITÄT BOCHUM RUB

Input:

```
eax 00000101 ebx 00000101 ecx 00000102 edx 00000103
esi 00000104 edi 00000105 ebp 00000106 esp 0013b4??
```

```
00000000000001111101000000011110110000000000000000000000001010101
```

Output:

```
eax 00000156 ebx 00000101 ecx 00000102 edx 00000103
esi 00000104 edi 00000105 ebp 00000106 esp 0013b4??
```

Early attempts

RUHR UNIVERSITÄT BOCHUM RUB

Input:

```
eax 00000101 ebx 00000101 ecx 00000102 edx 00000103
esi 00000104 edi 00000105 ebp 00000106 esp 0013b4??
```

```
00000000000001111101000000011110110000000000000000000000001010101
```

Output:

```
eax 00000156 ebx 00000101 ecx 00000102 edx 00000103
esi 00000104 edi 00000105 ebp 00000106 esp 0013b4??
```

eax = eax + 0x55

Early attempts

RUHR UNIVERSITÄT BOCHUM RUB

Input:

```
eax 00000101 ebx 00000101 ecx 00000102 edx 00000103
esi 00000104 edi 00000105 ebp 00000106 esp 0013b4??
```

```
00000000000001111101000000011110110000000000000000000000001010101
```

Output:

```
eax 00000156 ebx 00000101 ecx 00000102 edx 00000103
esi 00000104 edi 00000105 ebp 00000106 esp 0013b4??
```

eax = eax + 0x55

	Imm
00000000000001111101000000011110110000000000000000	0000000001010101
	0x55

Early attempts

RUHR UNIVERSITÄT BOCHUM RUB

Input:

```
eax 00000101 ebx 00000101 ecx 00000102 edx 00000103
esi 00000104 edi 00000105 ebp 00000106 esp 0013b4??
```

```
00000001100001111101000000011110110000000000000000000000011010101
```

Early attempts

RUHR UNIVERSITÄT BOCHUM RUB

Input:

```
eax 00000101 ebx 00000101 ecx 00000102 edx 00000103
esi 00000104 edi 00000105 ebp 00000106 esp 0013b4??
```

```
00000001100001111101000000011110110000000000000000000000011010101
```

Output:

```
eax 000001d4 ebx 00000101 ecx 00000102 edx 00000103
esi 00000104 edi 00000105 ebp 00000106 esp 0013b4??
```

Early attempts

RUHR UNIVERSITÄT BOCHUM RUB

Input:

```
  eax 00000101 ebx 00000101 ecx 00000102 edx 00000103
  esi 00000104 edi 00000105 ebp 00000106 esp 0013b4??
```

```
00000001100001111101000000011110110000000000000000000000001010101
```

Early attempts

RUHR UNIVERSITÄT BOCHUM RUB

Input:

```
eax 00000101 ebx 00000101 ecx 00000102 edx 00000103
esi 00000104 edi 00000105 ebp 00000106 esp 0013b4??
```

```
00000001100001111101000000011110110000000000000000000000001010101
```

Output:

```
eax 00000154 ebx 00000101 ecx 00000102 edx 00000103
esi 00000104 edi 00000105 ebp 00000106 esp 0013b4??
```

Early attempts

RUHR UNIVERSITÄT BOCHUM RUB

Input:

```
eax 00000101 ebx 00000101 ecx 00000102 edx 00000103
esi 00000104 edi 00000105 ebp 00000106 esp 0013b4??
```

```
00000001100001111101000000011110110000000000000000000000001010101
```

Output:

```
eax 00000154 ebx 00000101 ecx 00000102 edx 00000103
esi 00000104 edi 00000105 ebp 00000106 esp 0013b4??
```

$\text{eax} = \text{eax} \oplus 0\text{x}55$

Early attempts

RUHR UNIVERSITÄT BOCHUM RUB

Input:

```
eax 00000101 ebx 00000101 ecx 00000102 edx 00000103
esi 00000104 edi 00000105 ebp 00000106 esp 0013b4??
```

```
00000001100001111101000000011110110000000000000000000000001010101
```

Output:

```
eax 00000154 ebx 00000101 ecx 00000102 edx 00000103
esi 00000104 edi 00000105 ebp 00000106 esp 0013b4??
```

eax = eax $\oplus$ 0x55

```
Uk1 Operation                                                   Imm
0   000000110 00011111010000000111101100000000000000 0000000001010101
    xor                                                         0x55
```

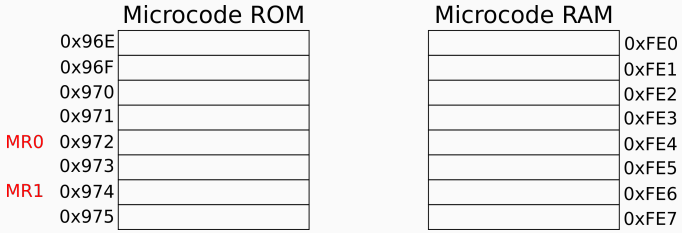
Micro Op Encoding

RUHR UNIVERSITÄT BOCHUM

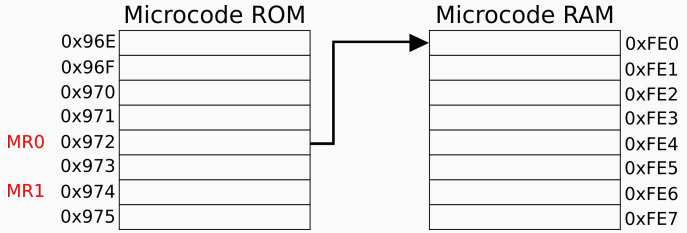
RUB

```
Uk1 Operation SwapOps OpMode Op1    Uk2 PZSFlags CFlag Uk3 OpClass SegReg Size Op2    RegMode Uk4    Uk5Imm Imm
u   ooooooooo x       m      111111 uuu f         f     u   CCC     ssss   zzz  222222 r       uuuuuu u      iiiiiiiiiiiiiiii
0   001111100 0       1      011111 010 0         0     0   000     1111   011  010110 0       000000 0      0000000011010101
    div2                     t24q                             reg     os4    64b  t15q                       0xd5

Uk1 Operation SwapOps OpMode Op1    Uk2 PZSFlags CFlag Uk3 OpClass SegReg Size Op2    RegMode Uk4    Uk5Reg Op3    Uk6Reg
u   ooooooooo x       m      111111 uuu f         f     u   CCC     ssss   zzz  222222 r       uuuuuu uu     333333 uuuuuuuuu
0   001111111 1       0      101001 100 0         0     0   001     0111   010  101010 1       010000 00     010000 000000000
    ld                       regmd5                           ld      rs     32b  t35d                       t9d

Uk1 ShortOprn Condition SwapOps OpMode Op1    Uk2 PZSFlags CFlag Uk3 OpClass SegReg Size Op2    RegMode Uk4    RomAddr
u   oooo      ccccc     x       m      111111 uuu f         f     u   CCC     ssss   zzz  222222 r       uuuuuu aaaaaaaaaaaaaaaa
0   0101      00100     1       1      111001 101 0         0     0   000     1111   011  111011 0       000000 00000000000000011
    jcc       EZF                      t50q                             reg     os4    64b  t52q                  0x3

Uk1              Action Uk2 RomAddr
uuuuuuuuuuuuuuuu ooo    uu  aaaaaaaaaaaa
1111111111111110 010    10  010110100101
                 branch     0x5a5
```

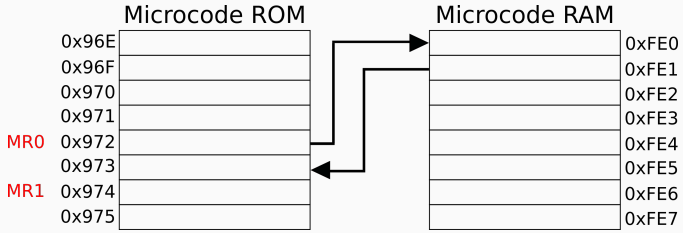
Infer Logic of ROM Triads



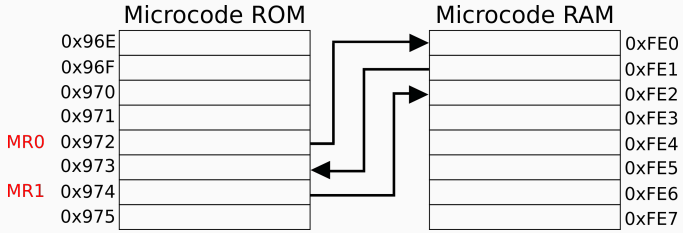
Infer Logic of ROM Triads

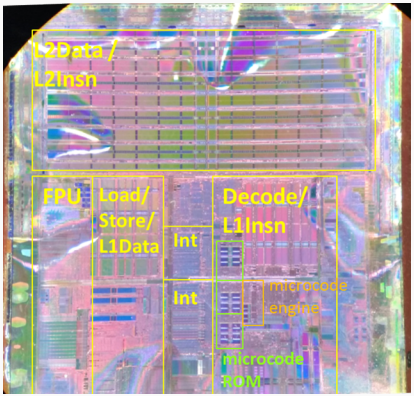


Infer Logic of ROM Triads



Infer Logic of ROM Triads

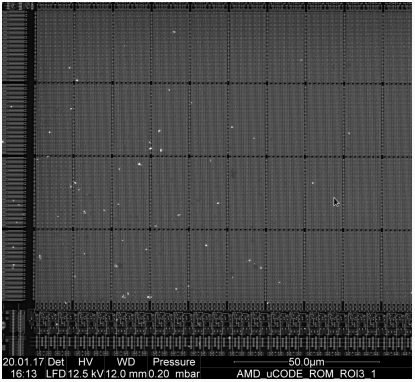



L2Data /
L2Insn
FPU
Load/
Store/
L1Data
Int
Int
Decode/
L1Insn
microcode
engine
microcode
ROM

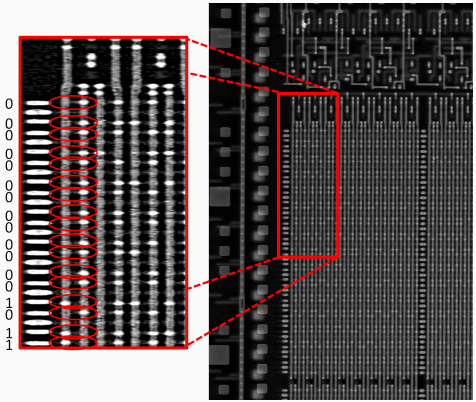
Hardware Analysis

RUHR UNIVERSITÄT BOCHUM

RUB



Hardware Analysis



RE Results

- Heatmaps

RE Results

- Heatmaps
- 29 Micro Ops
  - Logic, arithmetic, load, store
  - Write x86 program counter
  - Conditional microcode branch

RE Results

- Heatmaps
- 29 Micro Ops
  - Logic, arithmetic, load, store
  - Write x86 program counter
  - Conditional microcode branch
- Sequence word
  - Next triad, sequence complete, unconditional branch

RE Results

- Heatmaps
- 29 Micro Ops
  - Logic, arithmetic, load, store
  - Write x86 program counter
  - Conditional microcode branch
- Sequence word
  - Next triad, sequence complete, unconditional branch
- Substitution engine

Augment x86 instructions

- Jump back to ROM
  - `DIV`
- Emulate instruction logic
  - `IMUL, SHRD, CMPXCHG, ENTER`

Microprograms

- Instrumentation

Microprograms

- Instrumentation
- Remote microcode attacks
  - Control flow hijack in browsers induced by microcode
  - Triggered remotely with ASM.JS, WebAssembly

Microprograms

- Instrumentation
- Remote microcode attacks
  - Control flow hijack in browsers induced by microcode
  - Triggered remotely with ASM.JS, WebAssembly
- Cryptographic microcode Trojans
  - Introduce timing side-channels in constant-time ECC implementation
  - Inject faults to enable fault attacks

Sample Microprogram (simplified)

RUHR UNIVERSITÄT BOCHUM

RUB

```
sub.Z t1d, eax
jcc EZF, 0x2
or t12d, eax, 0x8
```

Sample Microprogram (simplified)

RUHR UNIVERSITÄT BOCHUM

RUB

```
sub.Z t1d, eax
jcc EZF, 0x2
or t12d, eax, 0x8

div2 t15q, t24q, 0xd5
srl t13w, ax, 0x8
div1.C t19d, t12d, t56d
```

Sample Microprogram (simplified)

RUHR UNIVERSITÄT BOCHUM RUB

```
sub.Z t1d, eax
jcc EZF, 0x2
or t12d, eax, 0x8

div2 t15q, t24q, 0xd5
srl t13w, ax, 0x8
div1.C t19d, t12d, t56d

mov t9d, t9d, regmd4
add.EP t56d, edx, t56d
jcc True, -0x800
```

Sample Microprogram (simplified)

RUHR UNIVERSITÄT BOCHUM RUB

```
sub.Z t1d, eax
jcc EZF, 0x2
or t12d, eax, 0x8

div2 t15q, t24q, 0xd5
srl t13w, ax, 0x8
div1.C t19d, t12d, t56d

mov t9d, t9d, regmd4
add.EP t56d, edx, t56d
jcc True, -0x800

mov eax, eax
add t1d, pcd, 1
writePC t1d
```

Outline

- What is microcode?
- Architectural crash course
- Is it hackable?
- Demo

Demo - Bug Attack Overview

RUHR UNIVERSITÄT BOCHUM RUB

- attack on implementation of otherwise secure crypto
- introduces error into calculation
- enables reconstruction of key material
- bug implemented via microcode update

Security issues

Security issues

- No signature, any update accepted

Security issues

- No signature, any update accepted
- Backdoors are possible

Security issues

- No signature, any update accepted
- Backdoors are possible
- Not really fixable (well, hardware recall...)

Security issues

- No signature, any update accepted
- Backdoors are possible
- Not really fixable (well, hardware recall...)
- Hacky fix: load update to brick update mechanism

Security issues

- No signature, any update accepted
- Backdoors are possible
- Not really fixable (well, hardware recall...)
- Hacky fix: load update to brick update mechanism
- But: requires strong attacker and old CPUs

Conclusion

RUHR UNIVERSITÄT BOCHUM RUB

- Microcode can be reversed and changed
- visit us at CCL 0, Multipurpose area ("Binary Security" in c3nav)!
- sample updates available on Github

`https://github.com/RUB-SysSec/Microcode`

Horst Görtz Institute for IT-Security
Ruhr-Universität Bochum

emproof
www.emproof.de