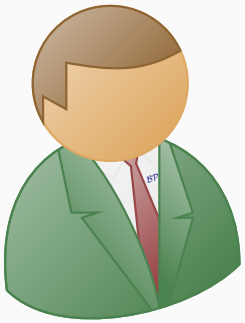
RISC-V

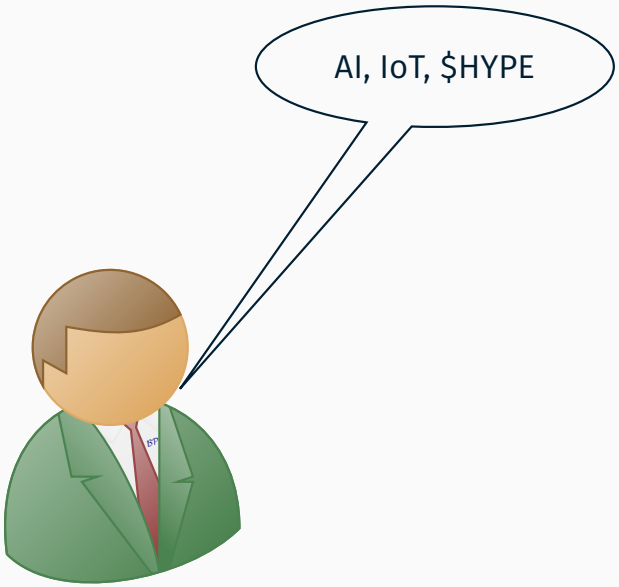
A new Hope

Hintergrund

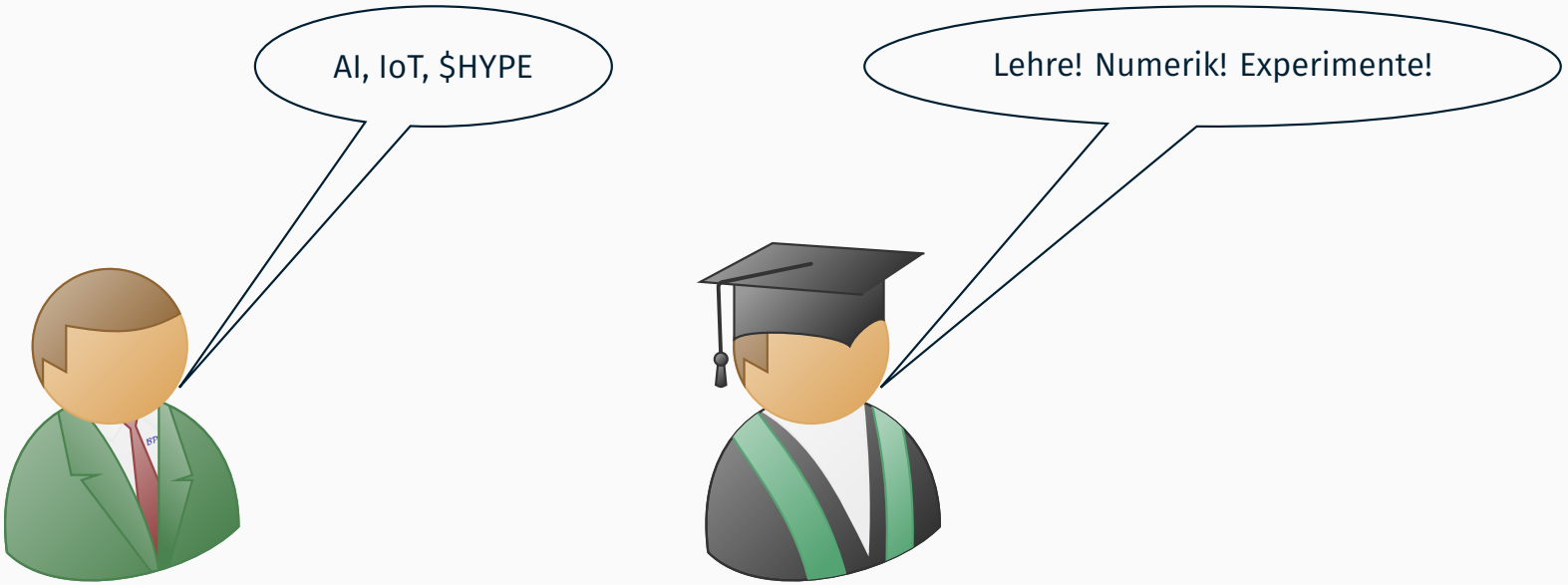
Eine CPU bauen …



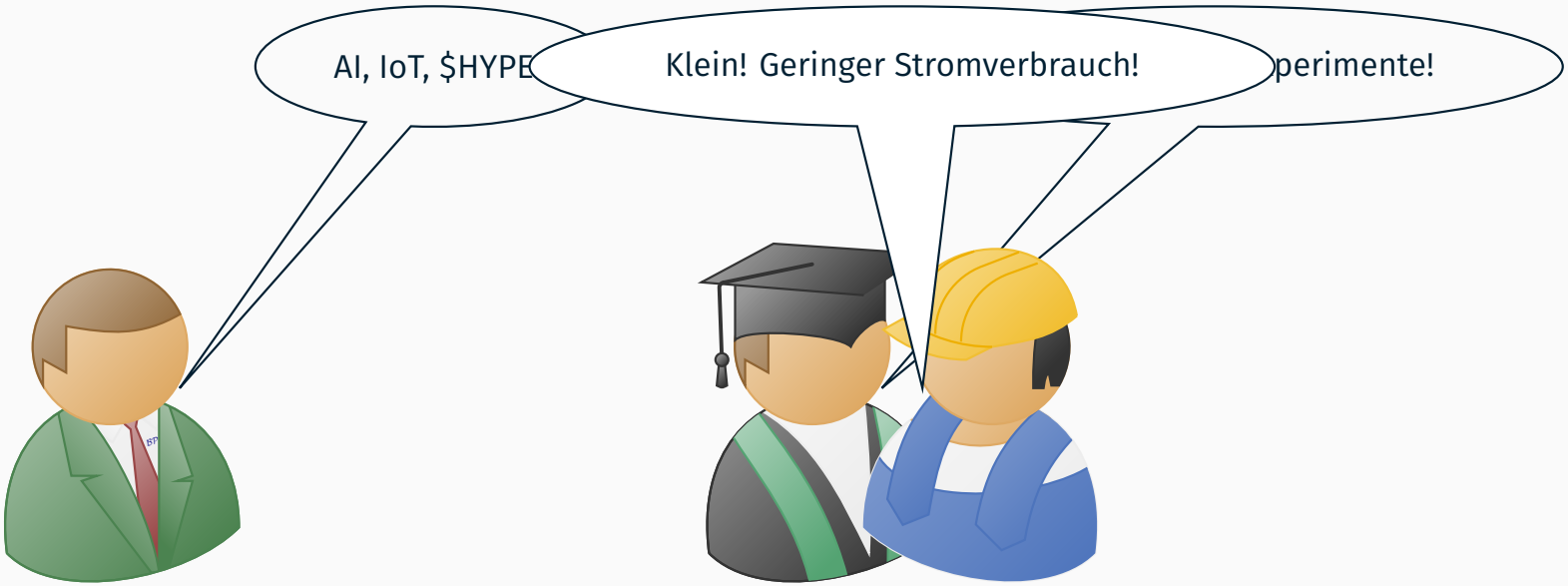
Eine CPU bauen …



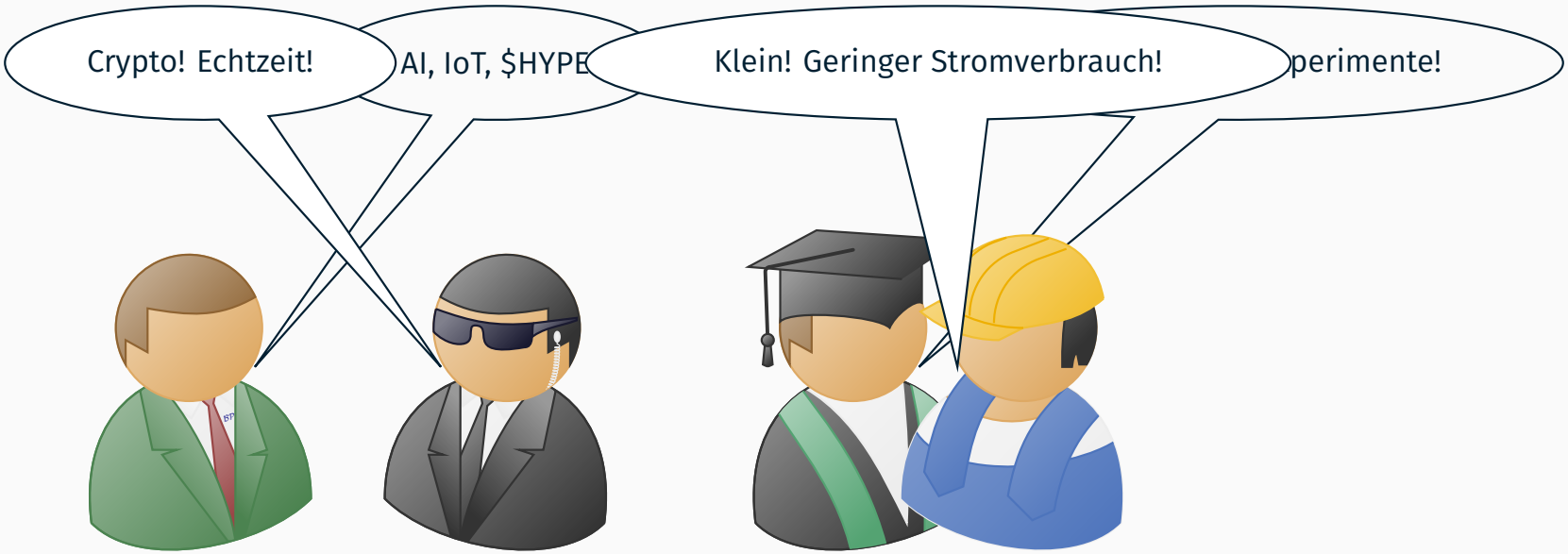
Eine CPU bauen …



Eine CPU bauen …



Eine CPU bauen …



Instruction Set Architecture (ISA)
oder
Die Programmiersprache der CPUs

Instruction Set Architecture

Abstraktes Modell eines Computers. Definiert:

- Befehle (opcodes)
- Speichermodell
- Addressierungsmodi
- Zustände und Zustandswechsel der abstrakten Maschine
- etc ...

Wähle deine ISA

Wähle deine ISA

- x86, x86_64, amd64

Wähle deine ISA

- x86, x86_64, amd64
- PowerPC

Wähle deine ISA

- x86, x86_64, amd64
- ARM
- PowerPC

Wähle deine ISA

- x86, x86_64, amd64
- ARM
- PowerPC
- SPARC

Wähle deine ISA

- x86, x86_64, amd64
- ARM
- PowerPC
- SPARC
- MIPS

Wähle deine ISA

- x86, x86_64, amd64
- ARM
- AVR
- Itanium
- PDP-11
- PowerPC
- SPARC
- MIPS
- VAX
- Transputer

Wähle deine ISA

- x86, x86_64, amd64
- ARM
- AVR
- Itanium
- PDP-11
- PowerPC
- SPARC
- MIPS
- VAX
- Transputer

Reichlich Auswahl!

Intel (x86, x86_64, amd64)

Pro:

Contra:

Intel (x86, x86_64, amd64)

Pro:

- Verbreitet, viel Werkzeug, Doku, etc.

Contra:

Intel (x86, x86_64, amd64)

Pro:

- Verbreitet, viel Werkzeug, Doku, etc.

Contra:

- “Historisch gewachsen”

Intel (x86, x86_64, amd64)

Pro:

- Verbreitet, viel Werkzeug, Doku, etc.

Contra:

- “Historisch gewachsen”
- Komplex

Intel (x86, x86_64, amd64)

Pro:

- Verbreitet, viel Werkzeug, Doku, etc.

Contra:

- “Historisch gewachsen”
- Komplex
- Unflexibel

Intel (x86, x86_64, amd64)

Pro:

- Verbreitet, viel Werkzeug, Doku, etc.

Contra:

- “Historisch gewachsen”
- Komplex
- Unflexibel
- Rechtssituation

Ich baue jetzt eine x86_64-CPU.


intel
TARGET AQUIRED

“Historisch gewachsen”?

- 1978 - Intel 8086 (16bit)

“Historisch gewachsen”?

- 1978 - Intel 8086 (16bit)
- 1985 - Intel 80386 (32bit, paging)

“Historisch gewachsen”?

- 1978 - Intel 8086 (16bit)
- 1985 - Intel 80386 (32bit, paging)
- 1986 - Intel Pentium (SMP, APIC)

“Historisch gewachsen”?

- 1978 - Intel 8086 (16bit)
- 1985 - Intel 80386 (32bit, paging)
- 1986 - Intel Pentium (SMP, APIC)
- 2003 - Athlon 64 (64bit)

“Historisch gewachsen”?

- 1978 - Intel 8086 (16bit)
- 1985 - Intel 80386 (32bit, paging)
- 1986 - Intel Pentium (SMP, APIC)
- 2003 - Athlon 64 (64bit)
- ...

Komplex?

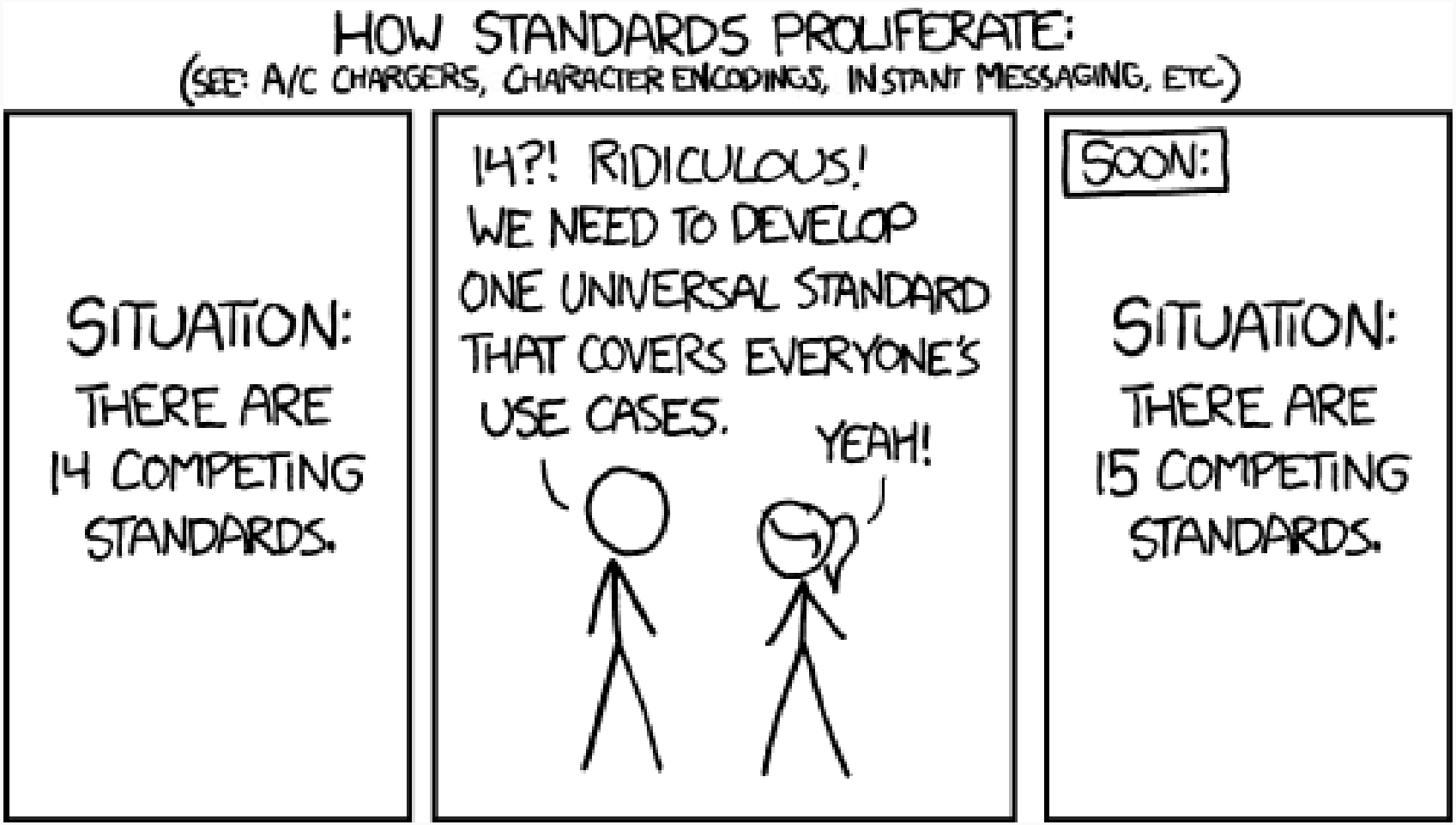
- 1000- >3000 Instruktionen[1]
- Handbuch: 5000 Seiten

[1] https://stefanheule.com/blog/how-many-x86-64-instructions-are-there-anyway/

Komplex?

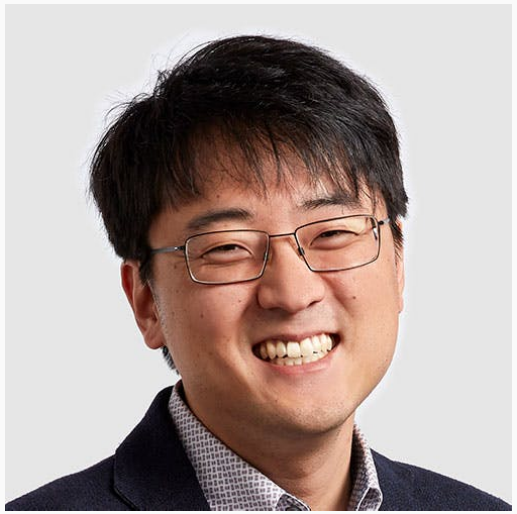
Neue ISAs braucht die Welt

Ein neuer Standard muss her!



https://xkcd.com/927/

RISC-V


Krste Asanovic, Yunsup Lee und Andrew Waterman (v.l. nach r.)

RISC-V - Der Plan

Eine Architektur für alle Bereiche:

RISC-V - Der Plan

Ein Rahmenwerk um ISAs zu konstruieren basierend auf einer gemeinsamen Grundlage

RISC-V - Der Plan

Ein Rahmenwerk um ISAs zu konstruieren basierend auf einer gemeinsamen Grundlage

- Wenige exklusive Basismodule, die grundlegende Strukturen definieren

RISC-V - Der Plan

Ein Rahmenwerk um ISAs zu konstruieren basierend auf einer gemeinsamen Grundlage

- Wenige exklusive Basismodule, die grundlegende Strukturen definieren
- Standardmodule für bekanntermassen häufig gebrauchte Operationen, sind optional

RISC-V - Der Plan

Ein Rahmenwerk um ISAs zu konstruieren basierend auf einer gemeinsamen Grundlage

- Wenige exklusive Basismodule, die grundlegende Strukturen definieren
- Standardmodule für bekanntermassen häufig gebrauchte Operationen, sind optional
- Ausgewiesener Platz für selbstgebaute Erweiterungen

RISC-V - Struktur

$=$

RV32E
RV32I
RV64I
RV128I

$= 1$

$+$

RVM
RVA
RVC
RVF/D/Q
Zicsr
DIY
...

≥ 0

Base extensions

Definiert:

Base extensions

Definiert:

- Registerlänge + -anzahl (16/32)

Base extensions

Definiert:

- Registerlänge + -anzahl (16/32)
- Addition/Subtraktion/Logische Operationen

Base extensions

Definiert:

- Registerlänge + -anzahl (16/32)
- Addition/Subtraktion/Logische Operationen
- Speicherzugriffe

Base extensions

Definiert:

- Registerlänge + -anzahl (16/32)
- Addition/Subtraktion/Logische Operationen
- Speicherzugriffe
- Branching

Base extensions

Definiert:

- Registerlänge + -anzahl (16/32)
- Addition/Subtraktion/Logische Operationen
- Speicherzugriffe
- Branching

$=\sim$ 50 Instruktionen

Standard extension: M

- Multiplikation
- Division
- Modulo

Standard extension: A

Atomare Speicheroperationen:

- *Load Reserverd/Store conditional*
- *Atomic SWAP/ADD/OR/...*

Standard extension: C

Komprimierte Instruktionen aus **I** mit 16bit Länge

- höhere Codedichte
- Instruktionen müssen seltener geladen werden/werden besser gezwischenspeichert

Standard extensions: F/D/Q

Floating Point Operationen

- mit 32 neuen Registern
- mit einer Länge von 32bit (F), 64bit (D), 128bit (Q)

Standard extension: Zicsr

- Bis 4096 **Control and status registers**
- Zugriff auf Zähler und Zeitnehmer
- Kontrolle von Systemfunktionen (Interrupts etc.)

Privilegien

User Mode
Supervisor Mode
Hypervisor Mode
Machine Mode
Debug Mode

Mögliche Kombinationen:

- M
- M,U
- M,S,U

H noch nicht fertig spezifiziert.

Wer macht das?

RISC-V Foundation

- Non-Profit
- Verwaltet die Weiterentwicklung
- Bald in der Schweiz (US-Exportgesetze)

Mitglieder


Google


SAMSUNG


NVIDIA


Western Digital


Alibaba Group

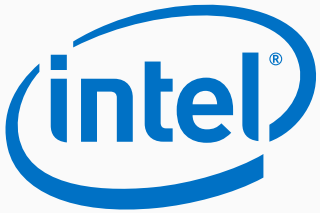

IBM


ETH zürich


Qualcomm

...

Nicht-Mitglieder



Wie probiere ich das aus?

Hardware

- Sifive[2]
  HiFive Unleashed - Linux-fähig (1000$)
- Kendryte K210[3]
  z.B. im **SiPEED MAIX**
- Ein paar (wenige) weiter...

[2] https://www.sifive.com/boards
[3] https://kendryte.com/

Softcores

Diverse freie Softcores[4]

[4] https://riscv.org/risc-v-cores/

Software[5]

- QEMU
- Linux
- GCC/clang/LLVM
- GDB
- FreeRTOS
- ...

[5] https://riscv.org/software-status/

Fragen?
Fragen!

ARTICLE

https://doi.org/10.1038/s41586-019-1493-8

Modern microprocessor built from complementary carbon nanotube transistors

Gage Hills[1,3], Christian Lau[1,3], Andrew Wright[1], Samuel Fuller[2], Mindy D. Bishop[1], Tathagata Srimani[1], Pritpal Kanhaiya[1], Rebecca Ho[1], Aya Amer[1], Yosi Stein[2], Denis Murphy[2], Arvind[1], Anantha Chandrakasan[1] & Max M. Shulaker[1]*

Electronics is approaching a major paradigm shift because silicon transistor scaling no longer yields historical energy–efficiency benefits, spurring research towards beyond–silicon nanotechnologies. In particular, carbon nanotube field–effect transistor (CNFET)–based digital circuits promise substantial energy–efficiency benefits, but the inability to perfectly control intrinsic nanoscale defects and variability in carbon nanotubes has precluded the realization of very–large–scale integrated systems. Here we overcome these challenges to demonstrate a beyond–silicon microprocessor built entirely from CNFETs. This 16–bit microprocessor is based on the RISC–V instruction set, runs standard 32–bit instructions on 16–bit data and addresses, comprises more than 14,000 complementary metal–oxide–semiconductor CNFETs and is designed and fabricated using industry–standard design flows and processes. We propose a manufacturing methodology for carbon nanotubes, a set of combined processing and design techniques for overcoming nanoscale imperfections at macroscopic scales across full wafer substrates. This work experimentally validates a promising path towards practical beyond–silicon electronic systems.

With diminishing returns of silicon field-effect transistor (FET) scaling[1], the need for FETs leveraging nanotechnologies has been steadily increasing. Carbon nanotubes (CNTs, nanoscale cylinders made of a single sheet of carbon atoms with diameters of approximately 10–20 Å) are prominent among a variety of nanotechnologies that are being considered for next-generation energy-efficient electronic systems[2–4]. Owing to the nanoscale dimensions and simultaneously high

CNFET gate, resulting in high leakage current and potentially incorrect logic functionality[17].

(2) Manufacturing defects. During wafer fabrication, CNTs inherently 'bundle' together, forming thick CNT aggregates[18,19]. These aggregates result in CNFET failure (reducing CNFET circuit yield), as well as prohibitively high particle contamination rates for very-large-scale integration (VLSI) manufacturing.