blindschleiche

Ein minimales Python für STM32F4-Discovery

Christoph Sieghart

Abstrakte Maschinen UE SS2014

Inhalt

- Überblick Python VM
- Überblick STM32F4-Discovery
- Development Setup
- Implementierung blindschleiche
- Features
- Demo

Python VM

- Stackmaschine

Python VM

- Stackmaschine
- Die Implementierung von CPython ist die Spezifikation

Python VM

- Stackmaschine
- Die Implementierung von CPython ist die Spezifikation
- .pyc Dateien sind gemarshallte `codeobjects`

Python VM

- Stackmaschine
- Die Implementierung von CPython ist die Spezifikation
- .pyc Dateien sind gemarshallte `codeobjects`
- VM führt fast direkt `codeobjects` aus

Python VM

- Stackmaschine
- Die Implementierung von CPython ist die Spezifikation
- .pyc Dateien sind gemarshallte `codeobjects`
- VM führt fast direkt `codeobjects` aus
- Python 2.7 hat eigene Bytecodes für `print`

STM32F4 Discovery

- Mikrocontroller **STM32F407VGT6**

STM32F4 Discovery

- Mikrocontroller **STM32F407VGT6**
  - ARM Cortex-M4 (32bit, ARMv7-M)

STM32F4 Discovery

- Mikrocontroller **STM32F407VGT6**
  - ARM Cortex-M4 (32bit, ARMv7-M)
  - 192KB RAM

STM32F4 Discovery

- Mikrocontroller **STM32F407VGT6**
  - ARM Cortex-M4 (32bit, ARMv7-M)
  - 192KB RAM
  - 1MB Flash

STM32F4 Discovery

- Mikrocontroller **STM32F407VGT6**
  - ARM Cortex-M4 (32bit, ARMv7-M)
  - 192KB RAM
  - 1MB Flash
- Mikrophon

STM32F4 Discovery

- Mikrocontroller **STM32F407VGT6**
  - ARM Cortex-M4 (32bit, ARMv7-M)
  - 192KB RAM
  - 1MB Flash
- Mikrophon
- 3-Achsen Beschleunigungssensor

STM32F4 Discovery

- Mikrocontroller **STM32F407VGT6**
  - ARM Cortex-M4 (32bit, ARMv7-M)
  - 192KB RAM
  - 1MB Flash
- Mikrophon
- 3-Achsen Beschleunigungssensor
- Audio DAC

STM32F4 Discovery

- Mikrocontroller **STM32F407VGT6**
  - ARM Cortex-M4 (32bit, ARMv7-M)
  - 192KB RAM
  - 1MB Flash
- Mikrophon
- 3-Achsen Beschleunigungssensor
- Audio DAC
- 4 LEDs, 2 Buttons

Development Setup

- OpenOCD und GDB

Development Setup

- OpenOCD und GDB
- **newlib** (malloc, ARM semi-hosting, . . .)

Development Setup

- OpenOCD und GDB
- **newlib** (malloc, ARM semi-hosting, . . .)
- CUBE4 von ST (HAL Schicht)

Development Setup

- OpenOCD und GDB
- **newlib** (malloc, ARM semi-hosting, . . .)
- CUBE4 von ST (HAL Schicht)
- Funktionelle Tests via fprintf und Python Skript

Implementierung blindschleiche

- CPython Interpreter übersetzt .py nach .pyc

Implementierung blindschleiche

- CPython Interpreter übersetzt .py nach .pyc
- .pyc Datei wird zum blindschleiche Binary gelinkt

Implementierung blindschleiche

- CPython Interpreter übersetzt .py nach .pyc
- .pyc Datei wird zum blindschleiche Binary gelinkt
- main() unmarshallt das .pyc `codeobject`

Implementierung blindschleiche

- CPython Interpreter übersetzt .py nach .pyc
- .pyc Datei wird zum blindschleiche Binary gelinkt
- main() unmarshallt das .pyc `codeobject`
- eval() führt Bytecode aus

Features 1

- Typen: `int`, `string`, `tuple`, `bool`, `None`

Features 1

- Typen: `int`, `string`, `tuple`, `bool`, `None`
- Integer Arithmetik

Features 1

- Typen: `int`, `string`, `tuple`, `bool`, `None`
- Integer Arithmetik
- lokale und globale Variablen

Features 1

- Typen: `int`, `string`, `tuple`, `bool`, `None`
- Integer Arithmetik
- lokale und globale Variablen
- Funktionen

Features 2

- `if` und `while`

Features 2

- `if` und `while`
- `print` via fprintf (ARM semi-hosting)

Features 2

- `if` und `while`
- `print` via fprintf (ARM semi-hosting)
- I/O via `print »DEST`

Demos

Fragen?