Snek: Baby Python

Keith Packard
Principal Engineer
SiFive
keithp@keithp.com
@keith_x11
Lego Robotics Class

• 10 students ages 10-12
  – 3-4 years of Lego-based instruction already
• 1-2 TAs ages 13-18
  – Former students
• 2 teachers + 2 adult volunteers
• 3 languages
  – Logo on Macintosh with Lego control panel
  – Robolab on Lego RCX
  – Snek on Arduino and Snekboard
• Cooperative Environment
My First Lego Computer

- Apple ][ (6502!)
- Floppy Disk
- Plug-in controller box
- Lego, so many Lego
- Logo Language
Logo Code

forever [
  talkto 1
  onfor 10
  sleep 10
]

Arduino

- 8-bit Atmega 328
- C++ language
- IDE on desktop
Arduino Code

```cpp
void setup() {
  pinMode(LED_BUILTIN, OUTPUT);
}

void loop() {
  digitalWrite(LED_BUILTIN, HIGH);  // turn the LED on (HIGH is the voltage level)
  delay(1000);                      // wait for a second
  digitalWrite(LED_BUILTIN, LOW);   // turn the LED off by making the voltage LOW
  delay(1000);                      // wait for a second
}
```

This example code is in the public domain.

http://www.arduino.cc/en/Tutorial/Blink
Fireworks

- Duemilanove
- SPI-based LED driver
- PWM hundreds of LEDs
Project Goals

- Python-inspired syntax, BASIC-inspired scale
- Run on a Duemilanove
- Small enough to explore in a few hours
Snek

- Small SoC
  - ATmega 328P
  - RISC-V
  - ARM Cortex-M
- Python-compatible
- IDE on desktop
  - sneakde
  - mu
Snek Line Bug

while True:
    talkto(M1)
    setright()
    on()
    while read(A1) > 0.35:
        pass
    off()
    talkto(M3)
    setleft()
    on()
    while read(A1) < 0.35:
        pass
    off()
Circuit Python

- Based on Micro Python
- Larger language (objects, etc)

<table>
<thead>
<tr>
<th>text</th>
<th>data</th>
<th>bss</th>
<th>dec</th>
<th>hex filename</th>
</tr>
</thead>
<tbody>
<tr>
<td>64440</td>
<td>48</td>
<td>21080</td>
<td>85568</td>
<td>14e40 snek.elf</td>
</tr>
<tr>
<td>246912</td>
<td>964</td>
<td>7124</td>
<td>255000</td>
<td>3e418 circuit-python.elf</td>
</tr>
</tbody>
</table>

- More sophisticated GPIO usage
import time
import board
from digitalio import DigitalInOut, Direction, Pull

led = DigitalInOut(board.D13)
led.direction = Direction.OUTPUT

switch = DigitalInOut(board.D2)
switch.direction = Direction.INPUT
switch.pull = Pull.UP

while True:
    if switch.value:
        led.value = False
    else:
        led.value = True

    time.sleep(0.01)  # debounce delay
Snek Example

talkto(M1)
while True:
    if read(A1):
        on()
    else
        off()
mu – a Python IDE

- An IDE for writing Python, written in Python.
- Support for MicroPython devices
- Easily extended to support Snek
- Merge Request pending upstream
Editing Snek with Mu

```python
1  talkto(M2)
2  while True:
3       onfor(1)
4       time.sleep(1)
```
Editing Snek with Snekde

```python
text from the image
```
Language Implementations

- Direct interpretation
  - BASIC, LISP
  - Source code is the executable

- Bytecode
  - Python, Snek, Perl, Ruby, Javascript, Java, ...
  - Compiler output runs on a virtual machine

- Compiled
  - Rust, C, Fortran
  - Compiler output runs directly on the CPU
Snek Values

- 32-bit IEEE float
- Tuple/List/Dictionary
- String
- Compiled Function
- Builtin Function
- True, False, None
NaN-o-Rama

- Exponent == 0xff → NaN
  - 16 million NaNs!
Snek Compiler

- LL(1) table-driven parser using Lola
- Hand-written lexer
- Old-school direct code generation
- About 1500 lines of code
Lola

- Parser generator (written in Python)
- Small parser
  - Lola: 3600 code + 1200 data
  - Bison: 4400 code + 6000 data

```plaintext
formal : NAME
    @{
        snek_parse_formals[snek_parse_nformal++] = snek_token_val.id;
    }@
    opt-named-p
```
while-stat:
@{
    /* push 0 - top_off */
    value_push_offset(snek_code_current());
}@ 

WHILE expr COLON
@{
    snek_code_add_op_offset(snek_op_branch_false, 0);
    /* push 1 - while_off */
    value_push_offset(snek_compile_prev);
}@ 
suite
@{
    /* push 2 - loop_end_off */
    snek_code_add_op_offset(snek_op_branch, 0);
    value_push_offset(snek_compile_prev);
    /* push 3 - while_else_stat_off */
    value_push_offset(snek_code_current());
}@ 

while-else-stat
@{
    patch_loop:;
    snek_offset_t while_else_stat_off = value_pop().offset;
    snek_offset_t loop_end_off = value_pop().offset;
    snek_offset_t while_off = value_pop().offset;
    snek_offset_t top_off = value_pop().offset;

    snek_code_patch_branch(while_off, while_else_stat_off);
    snek_code_patch_branch(loop_end_off, top_off);
    snek_code_patch_forward(while_off, loop_end_off, snek_forward_continue, top_off);
    snek_code_patch_forward(while_off, loop_end_off, snek_forward_break, snek_code_current());
};
Snek VM

- Stack with Accumulator
- 61 opcodes
  - 39 expression operators (+, -, =, …)
  - Python-specific operators like “slice”
- Non-recursive implementation
  - C stack stays within known bounds
Snek Bytecode

```python
> for i in range(10):
+    print("hello %d" % i)
+
```
Snek Memory Manager

- Mark & Sweep Garbage Collector
  - uses only one heap
- Compacting
  - smooshes allocated objects all together
- Incremental
  - frees unused objects in chunks
- Bounded Recursion
  - finite stack usage during collect
Memory Management
Tricky Python Bits

- Lexical white space
  - differs between REPL and files
  - error recovery is tricky
- Function Parameters
  - named vs positional
  - required vs optional
- Optimizing dictionaries
- Shared constants
  ```python
  >>> a = (1,2,3)
  >>> b = (1,2,3)
  >>> a is b
  False
  >>> (1,2,3) is (1,2,3)
  True
  ```
SnekBoard

- Drives Lego Power Functions motors
- Runs Snek and CircuitPython
- Crowd Supply campaign starting soon!

https://www.crowdsupply.com/keith-packard/snekboard
Demo