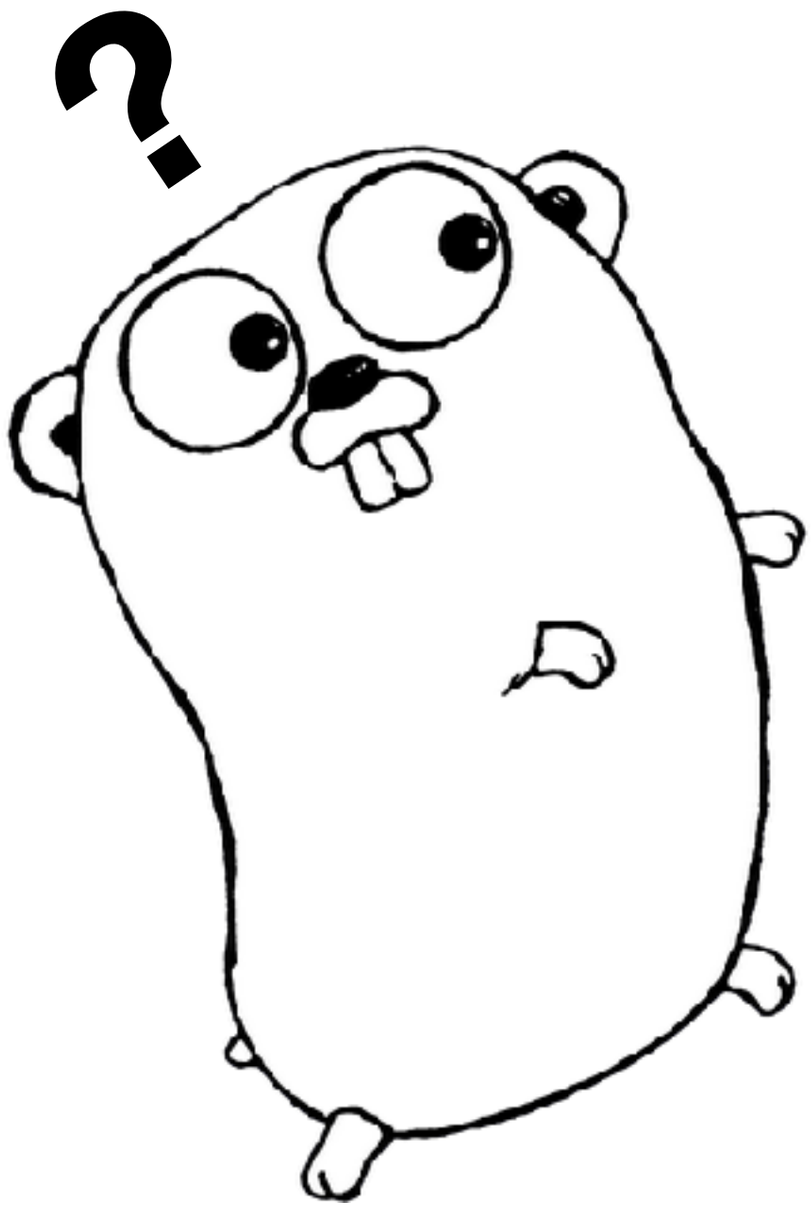
TypeScript Go(es) Embedded



@noctarius2k

Reacts to: Chris

@noctarius2k

Codes Java

@noctarius2k

Prefers Kotlin

Codes Java

@noctarius2k

Forced to use JS

Prefers Kotlin

Codes Java

@noctarius2k

Forced to use JS

Prefers Kotlin

Kinda likes Go

Codes Java

@noctarius2k

Forced to use JS

Prefers Kotlin

Kinda likes Go

Adores TypeScript

Codes Java

@noctarius2k

Forced to use JS

Prefers Kotlin

Kinda likes Go

Adores TypeScript

Codes Java

LOVES BEER!
(German, obviously)

@noctarius2k

@noctarius2k

Garbage Collection

@noctarius2k

Garbage Collection

Performance Optimization

@noctarius2k

Garbage Collection

Performance Optimization

JVM Internals

@noctarius2k

Disclaimer

Disclaimer

This Talk Is Very Subjective!

Disclaimer

This Talk Is Very Subjective!

I'm Very Opinionated! *

Disclaimer

This Talk Is Very Subjective!

I'm Very Opinionated! *

I Have A Lot Of Hate In Me ;-) *

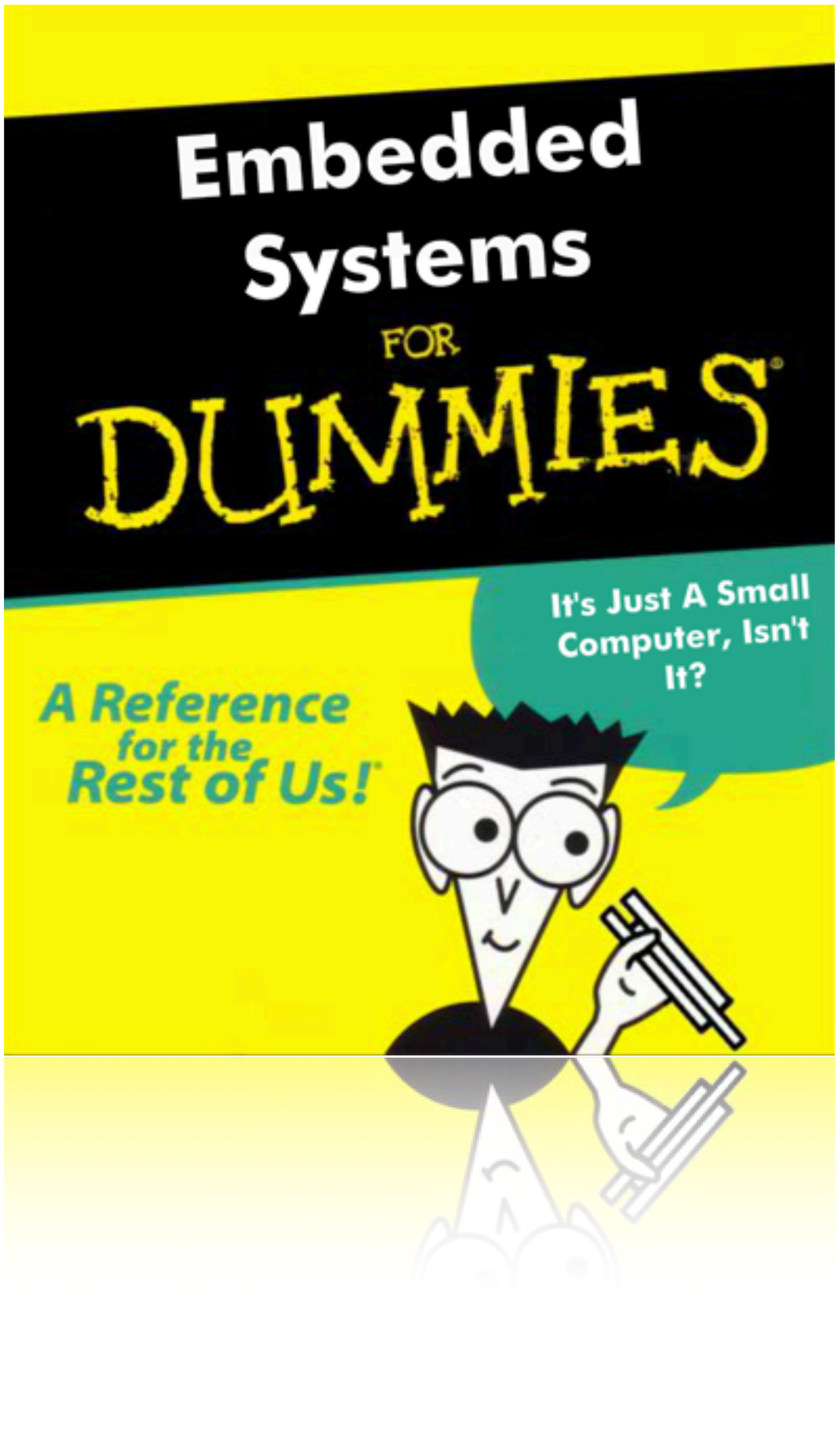
Disclaimer

This Talk Is Very Subjective!

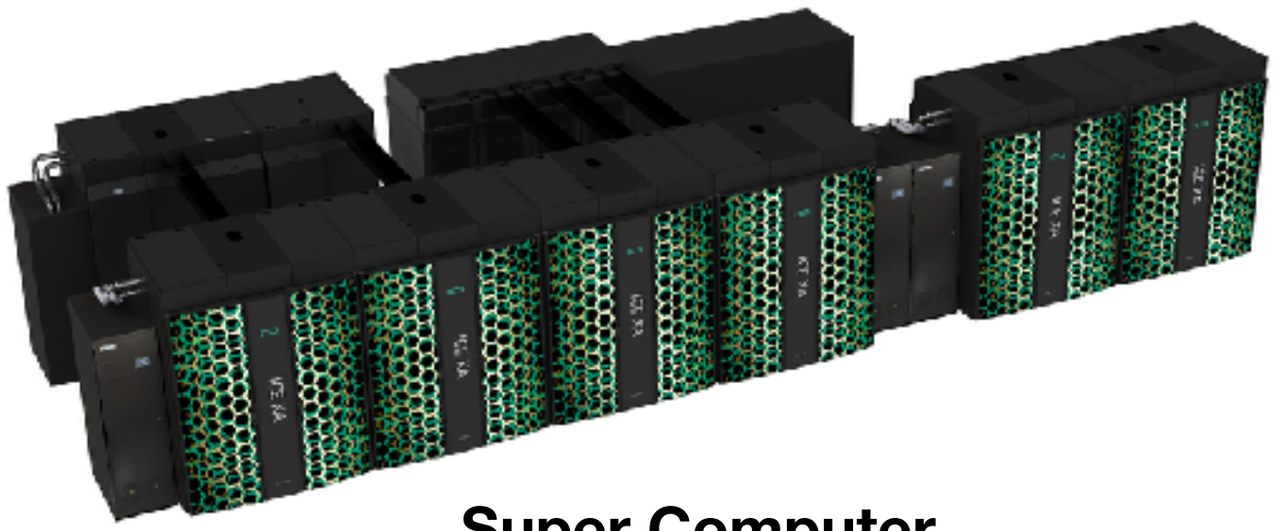
I'm Very Opinionated! *

I Have A Lot Of Hate In Me ;-) *

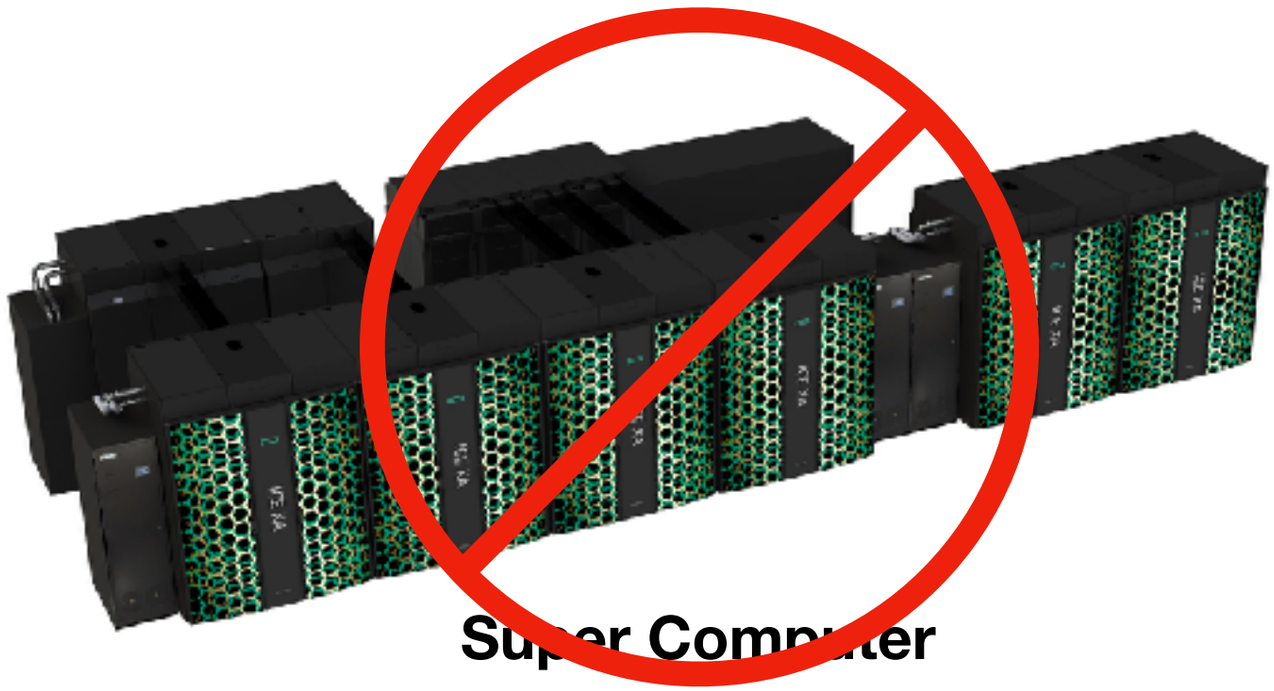
* Happy To Have Plenty Of Discussions!



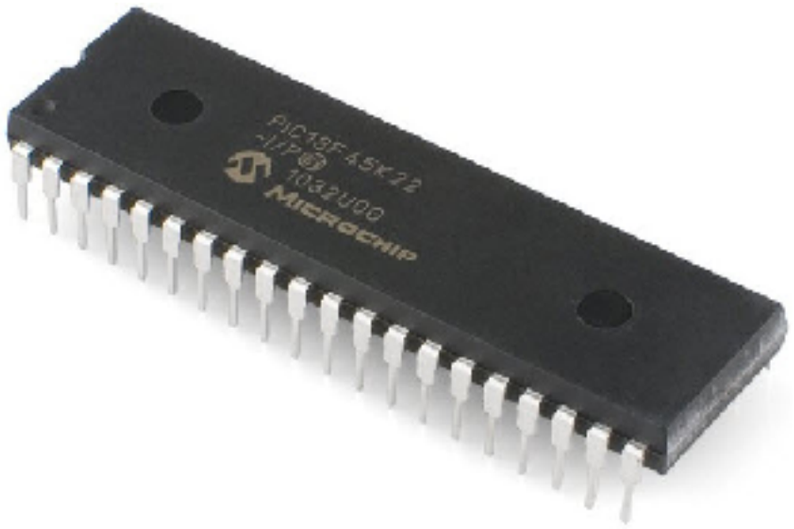
So what are we talking about?



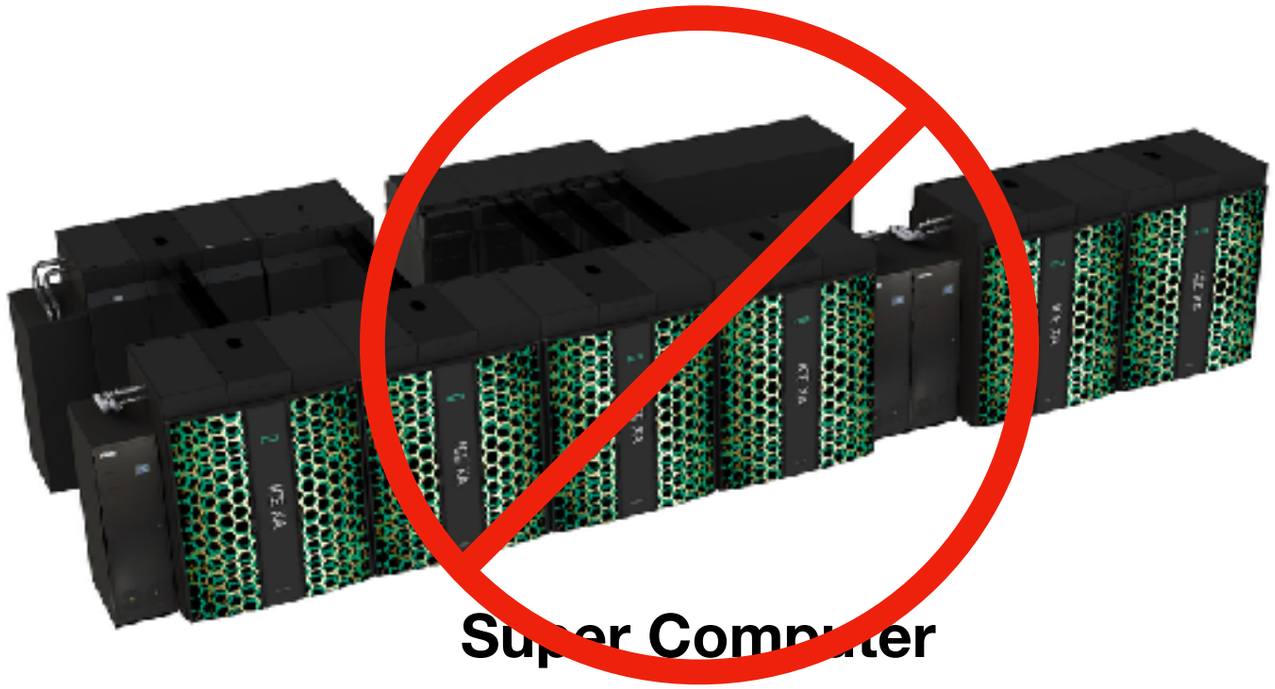
Super Computer



Super Computer



Microcontroller (PIC)



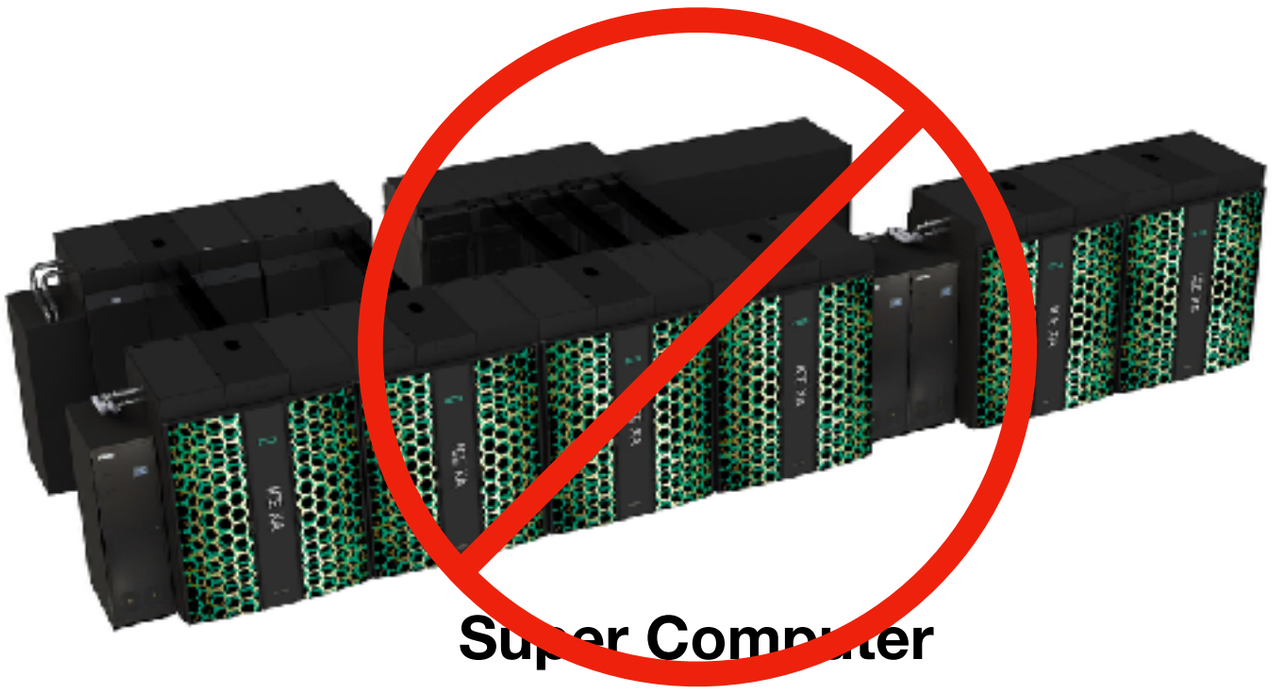
Super Computer

Microcontroller (PIC)

Super Computer

Microcontroller (PIC)

Desktop Computer (no special brand ;-))



Super Computer

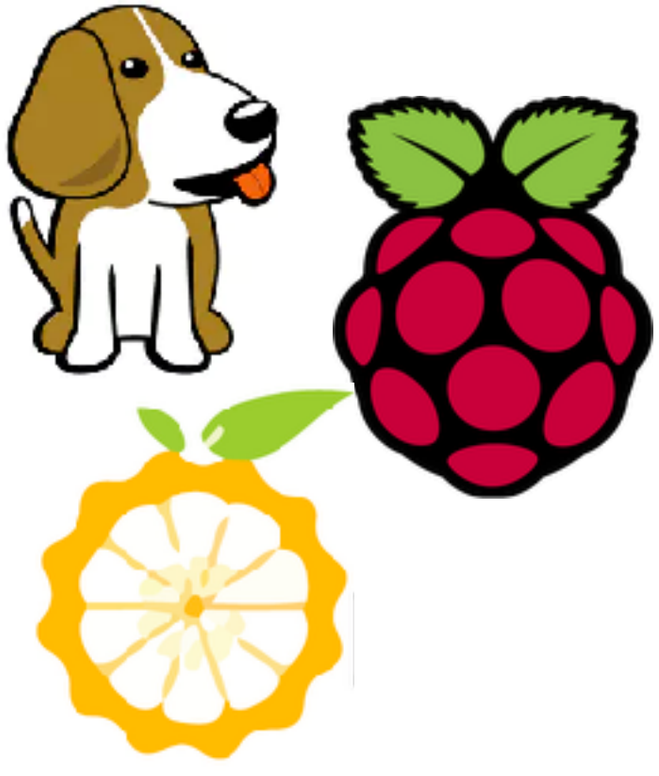
Microcontroller (PIC)

Desktop Computer (no special brand ;-))

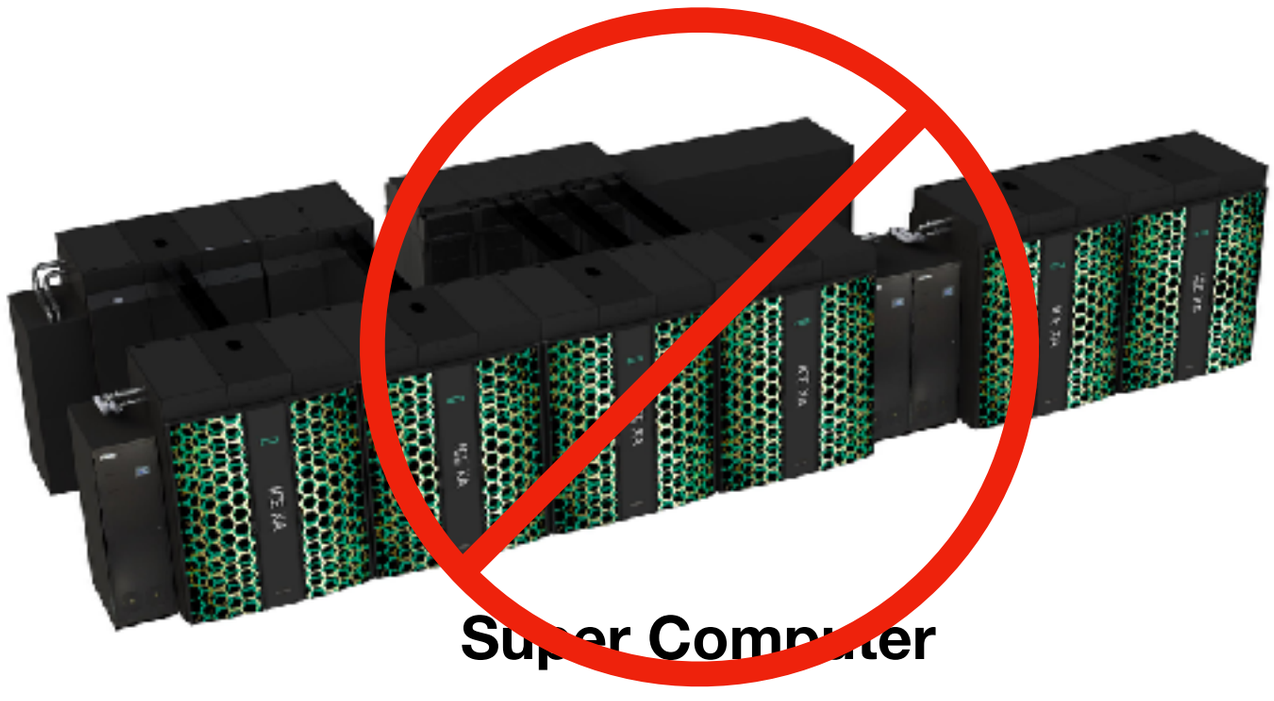
Super Computer

Microcontroller (PIC)

Desktop Computer (no special brand ;-))



Single Board Computer (SBC)

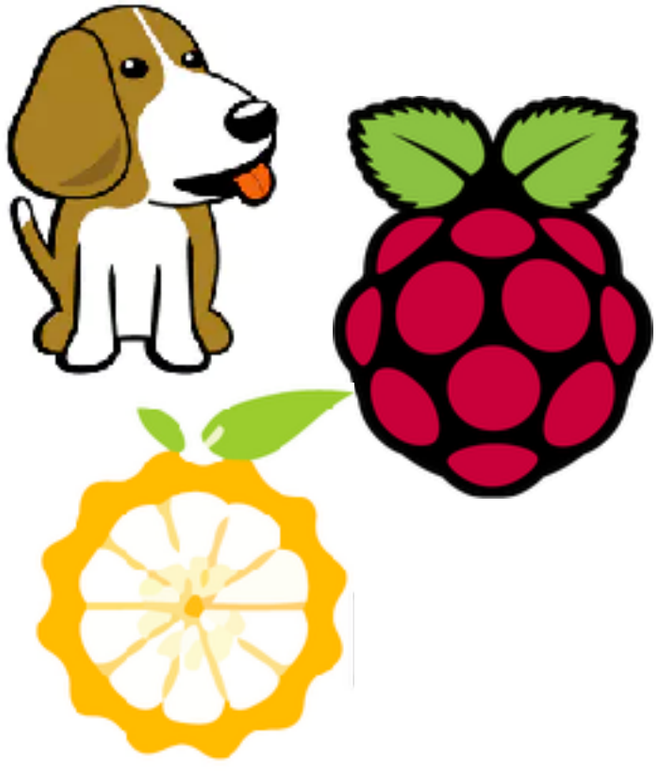


Super Computer

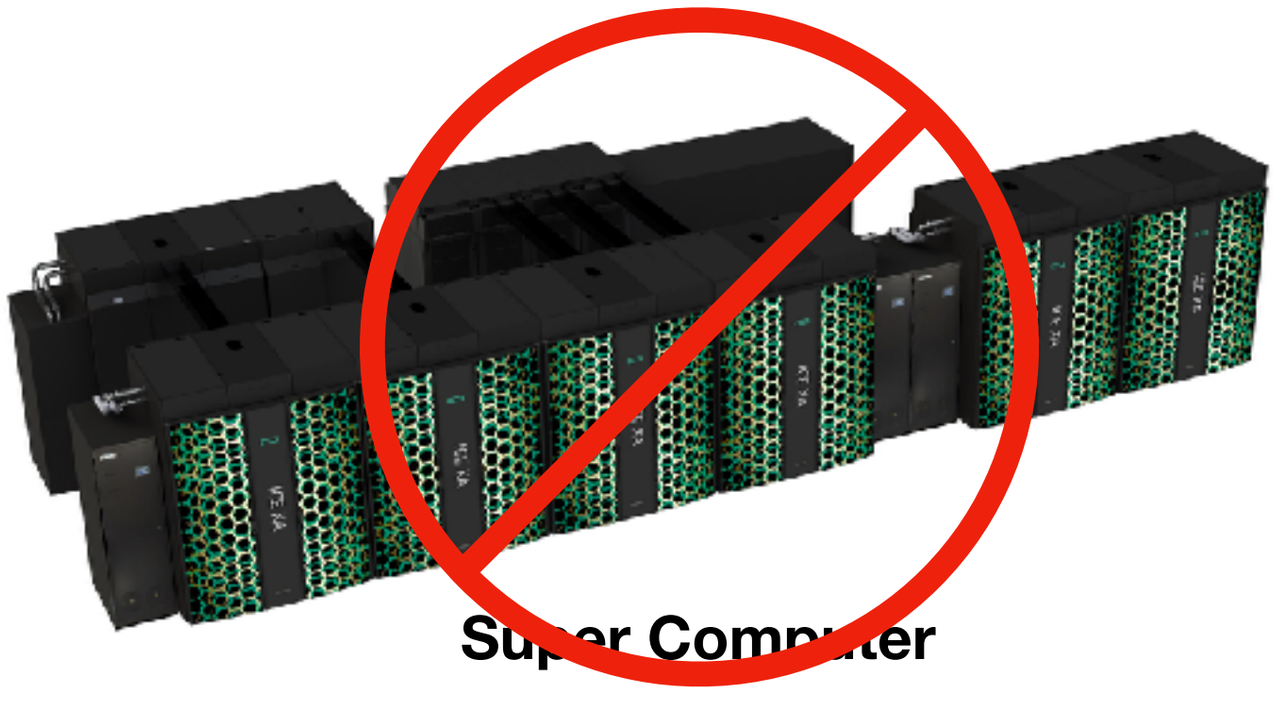
Microcontroller (PIC)

Desktop Computer (no special brand ;-))

!

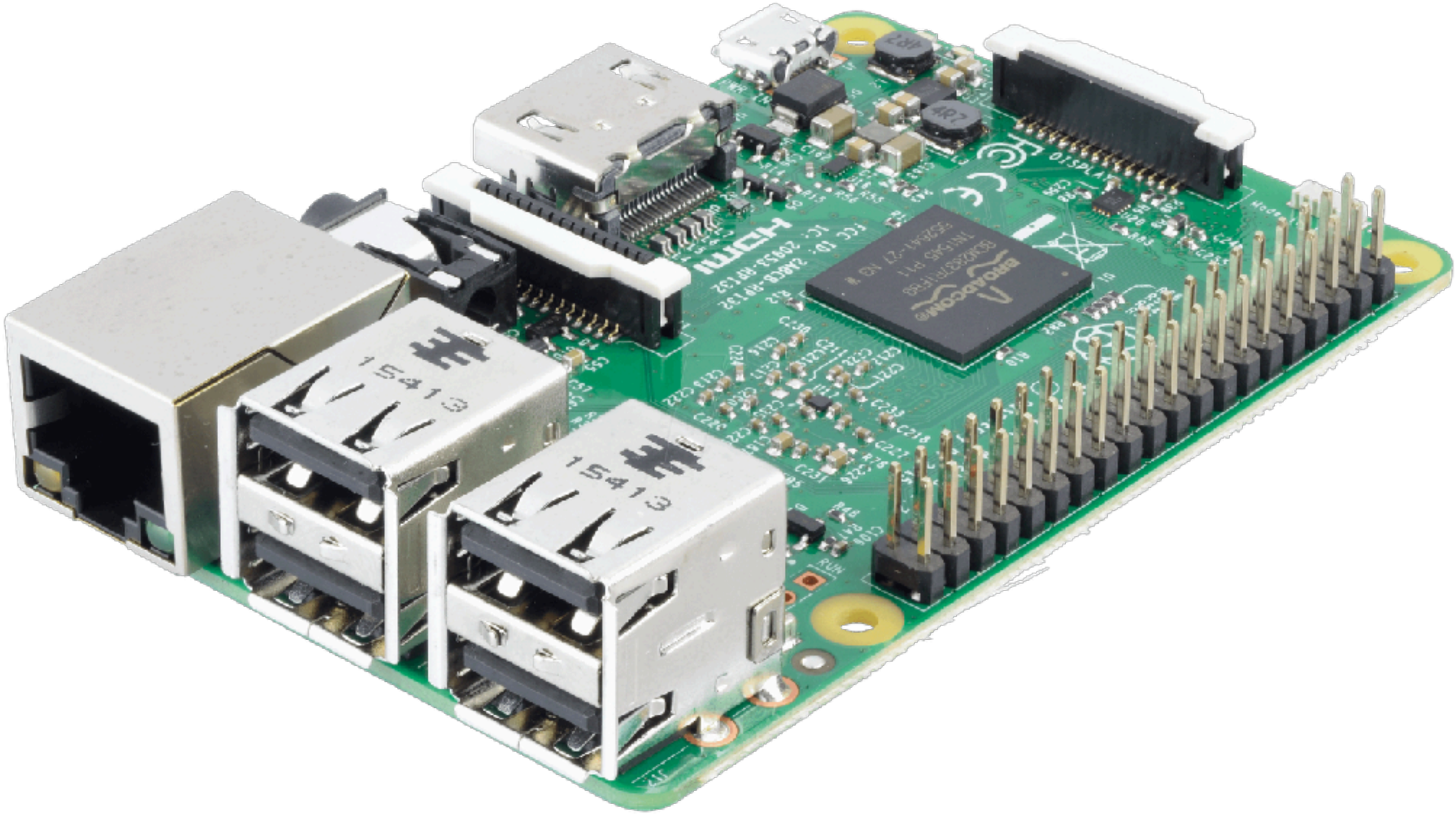


Single Board Computer (SBC)



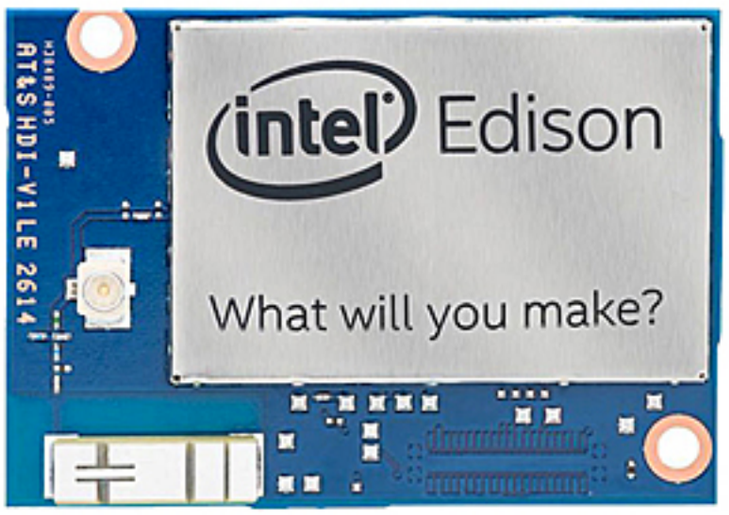
Super Computer

Probably the most known system of that kind

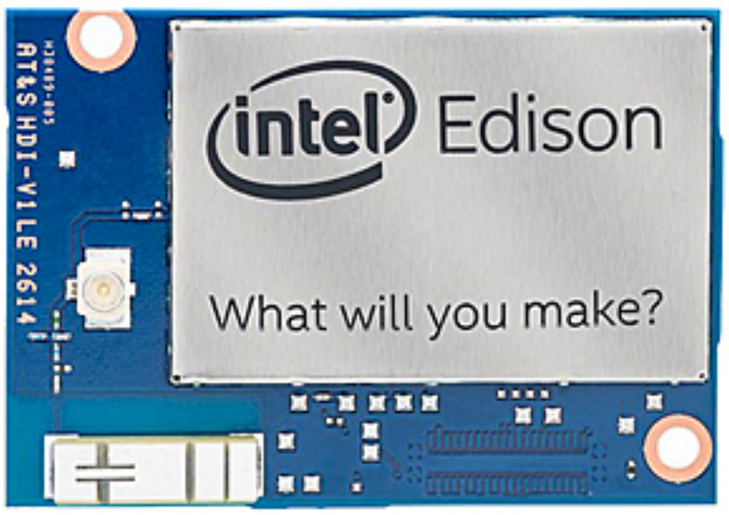


Different Types Of CPU Architectures

Different Types Of CPU Architectures

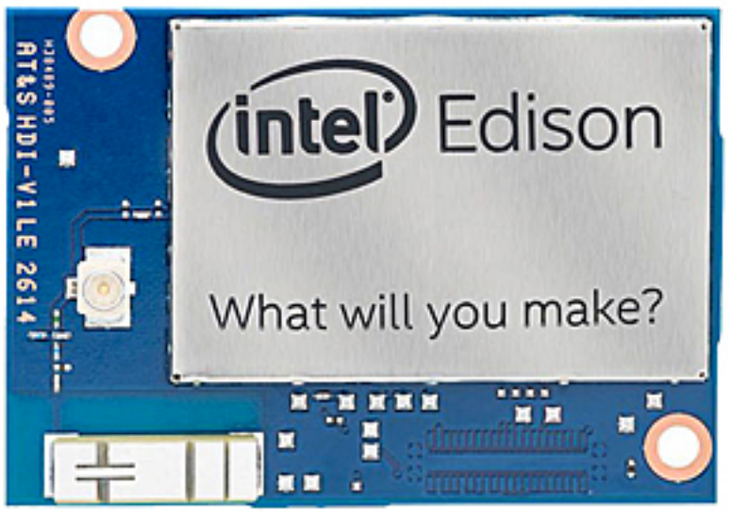


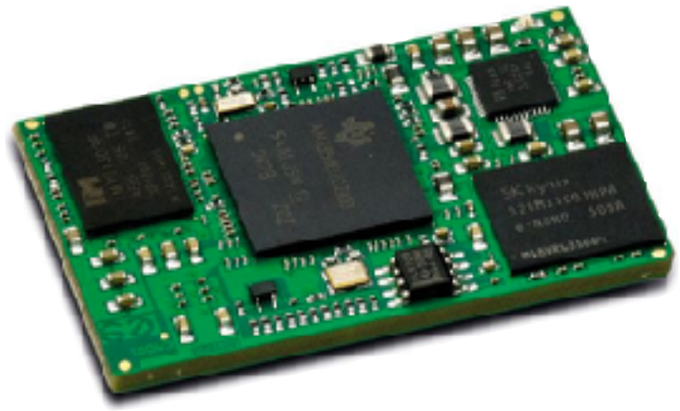
Different Types Of CPU Architectures



Different Types Of CPU Architectures

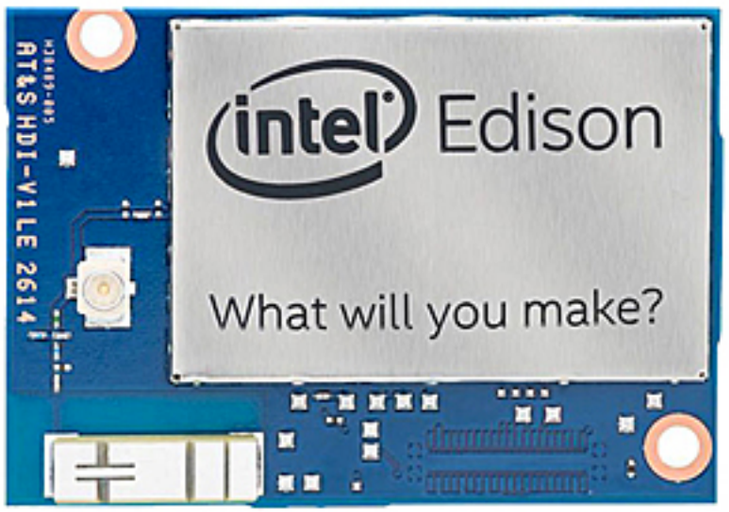
MIPS
TECHNOLOGIES

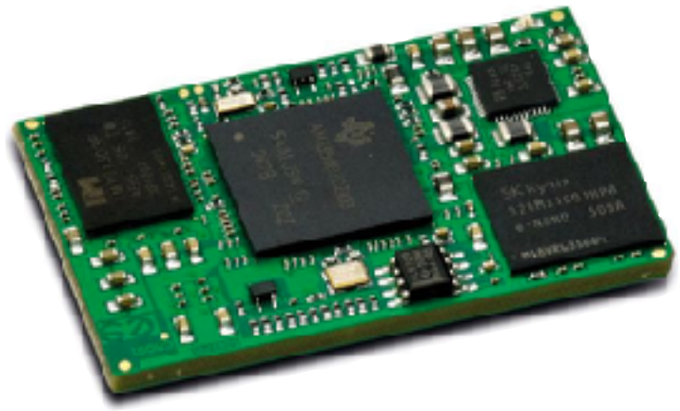




Different Types Of CPU Architectures

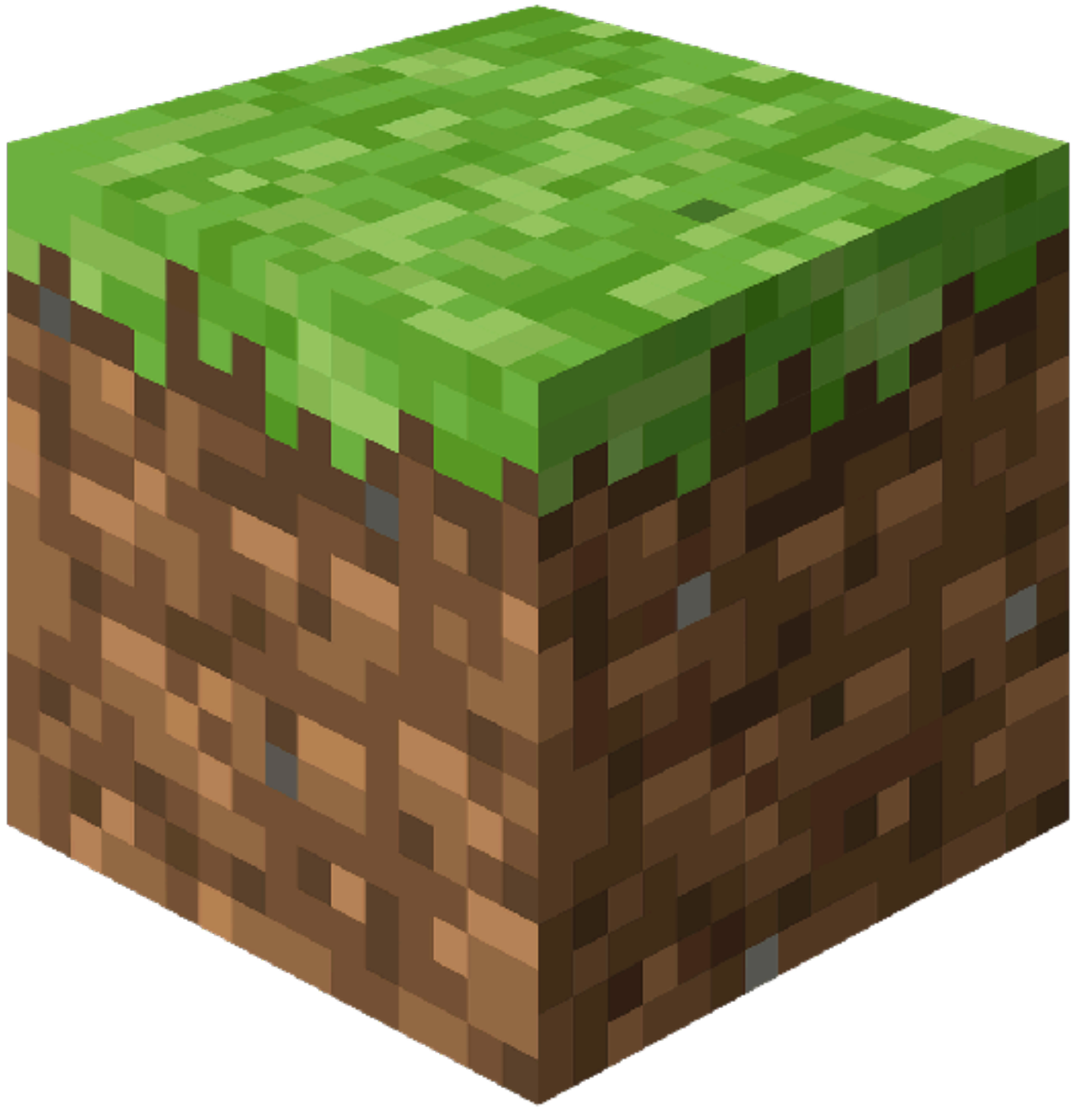
MIPS
TECHNOLOGIES



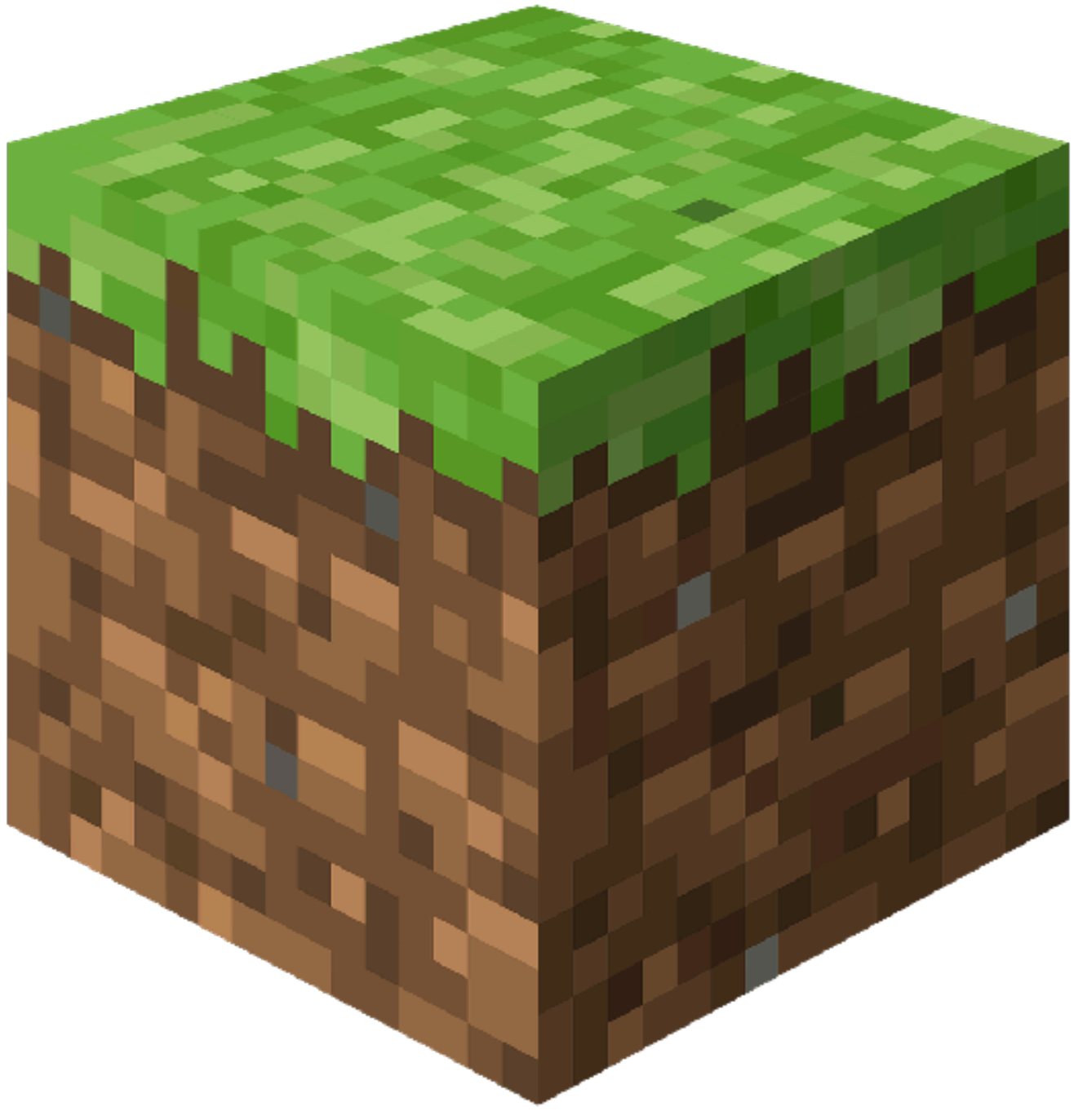


*** Plenty of different architecture variants (most common A7 / A8)**

And People Actively Build Products On Those SBCs

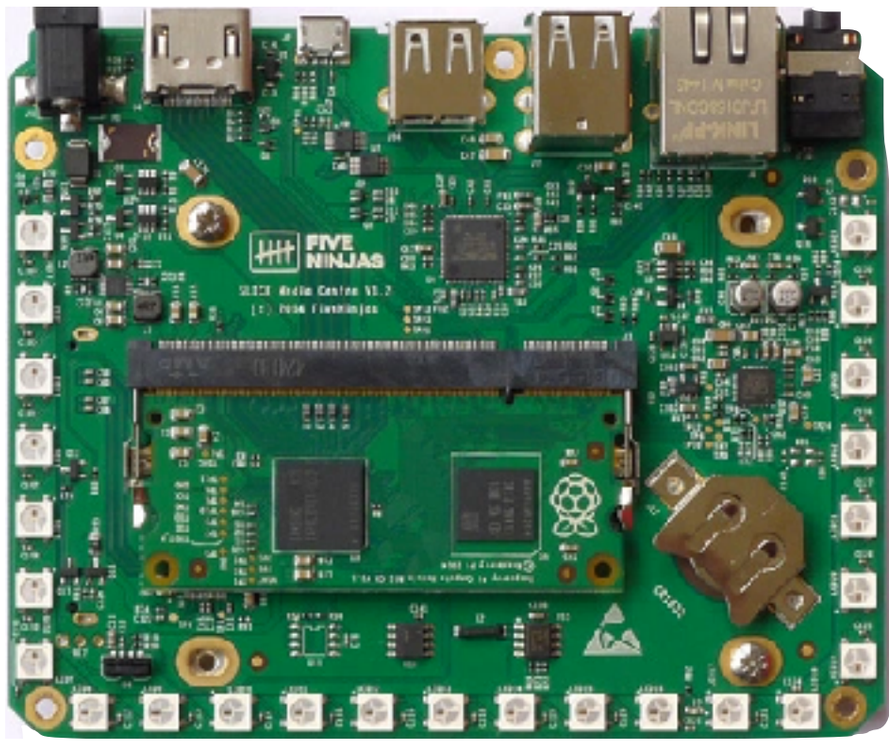


And People Actively Build Products On Those SBCs

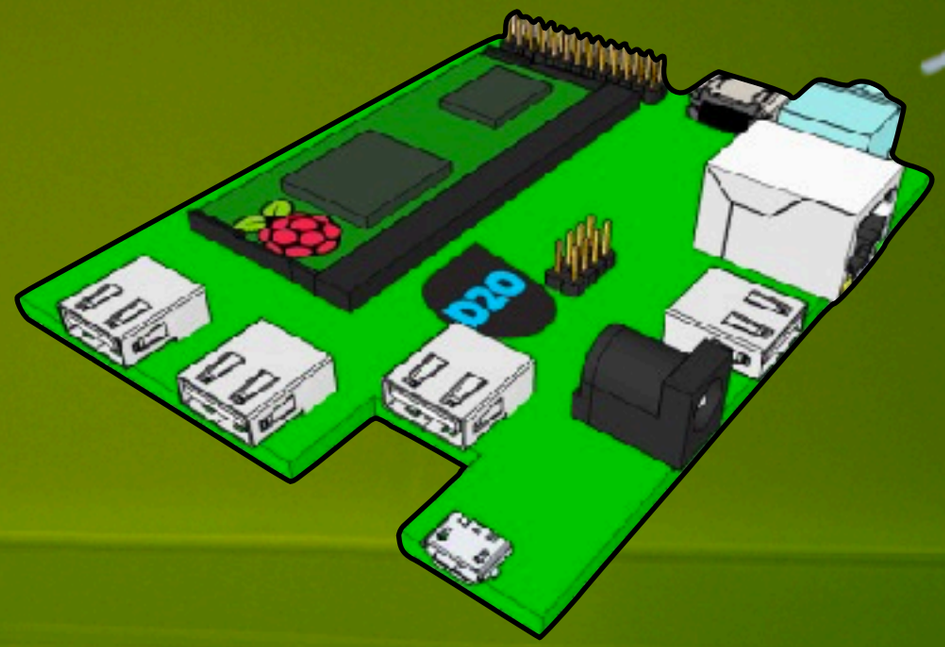


* Just a few quick RaspberryPi examples

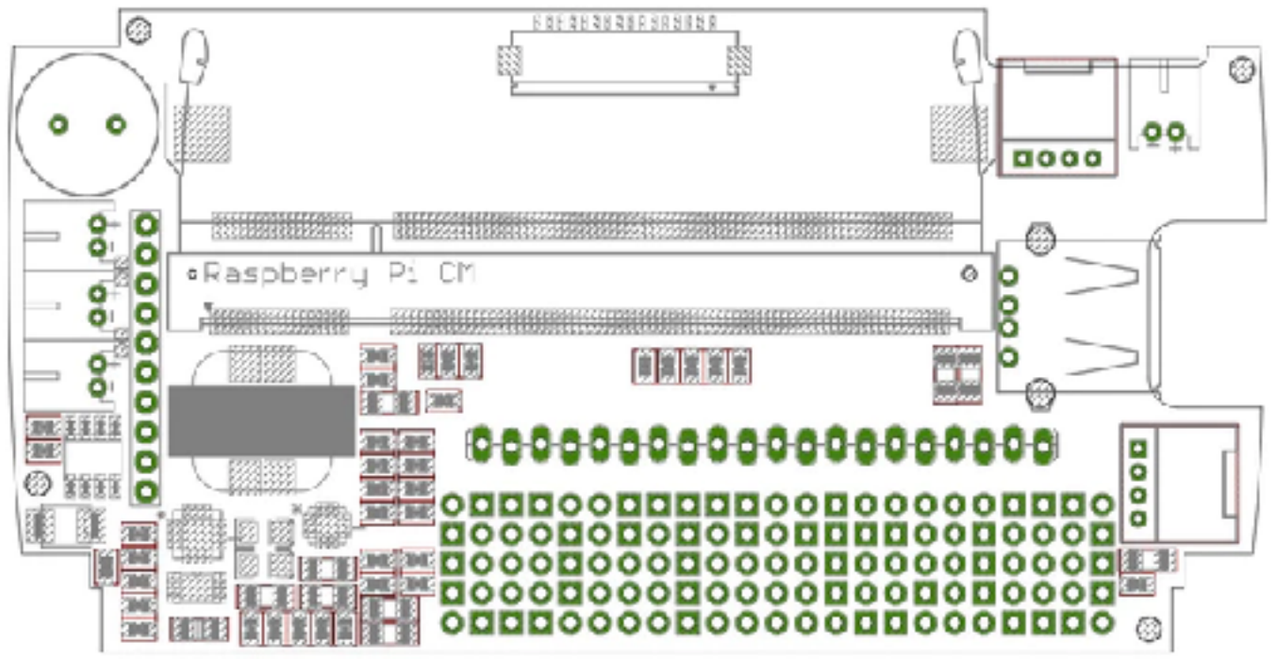
https://pi-top.com/



http://fiveninjas.com/

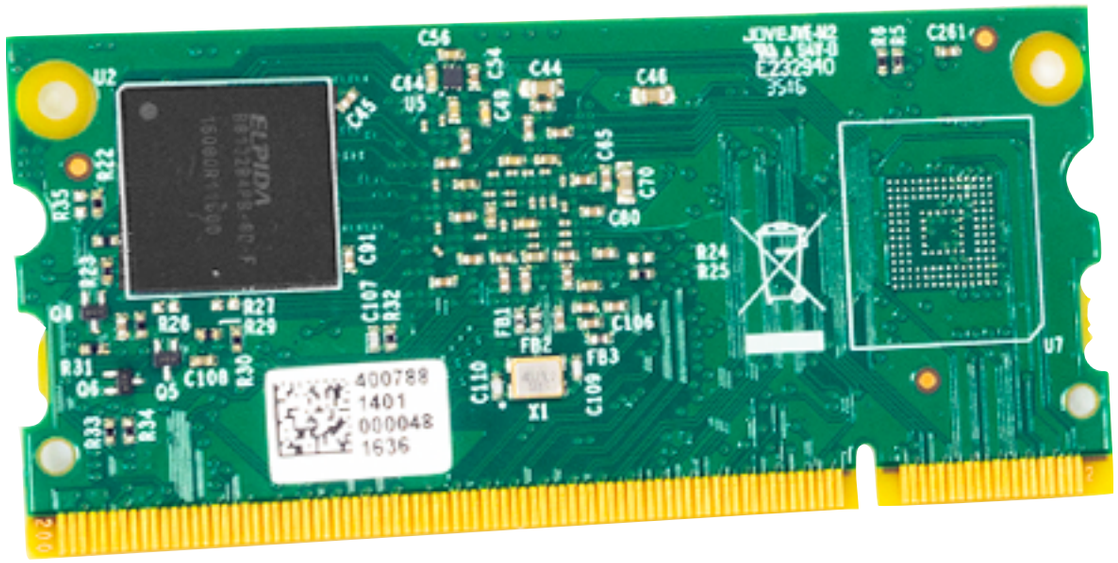


http://dock2office.com/raspberry-pi/

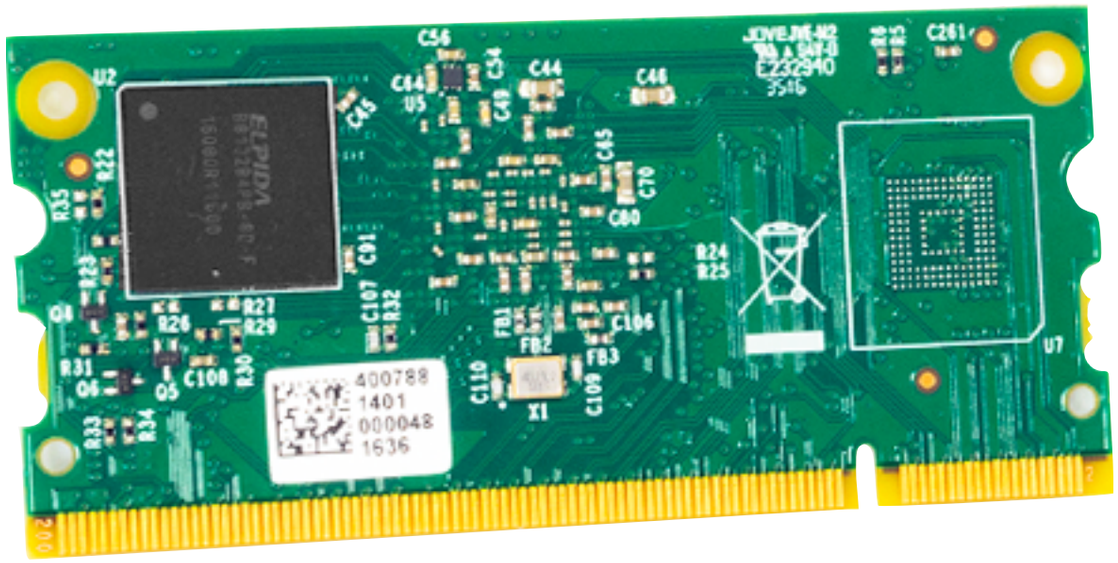


https://www.kickstarter.com/projects/1598272670/meet-otto-the-hackable-gif-camera

More Modules With Clear Focus On Commercialization

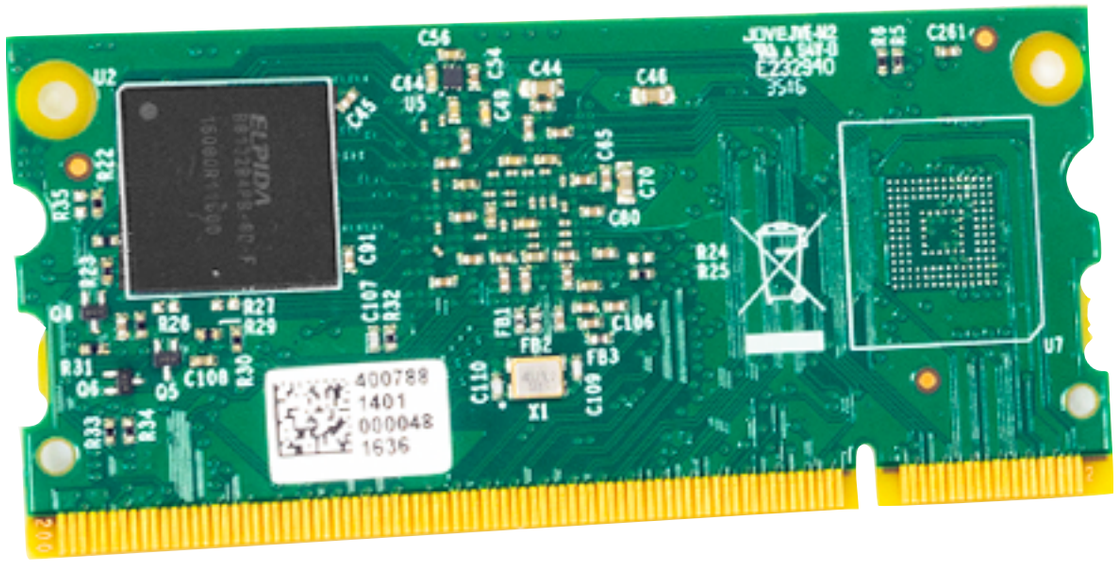


RaspberryPi Compute Module



RaspberryPi Compute Module

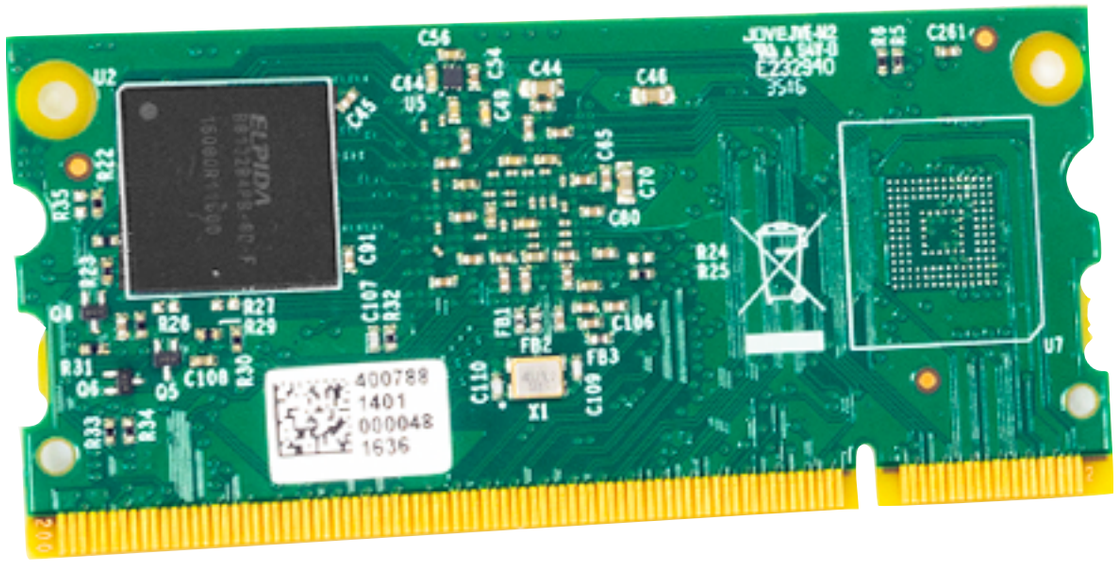
Chip Pro (not recommended)



RaspberryPi Compute Module

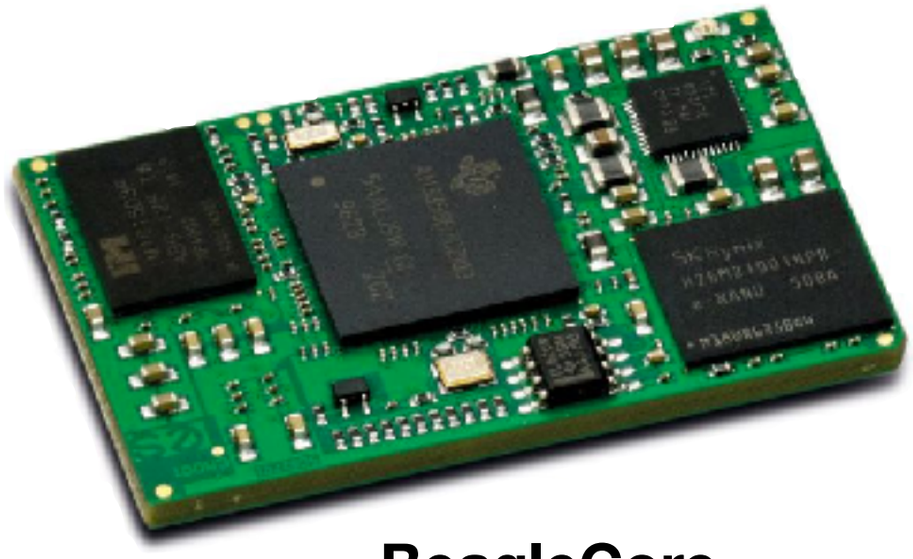
Omega 2s

Chip Pro (not recommended)



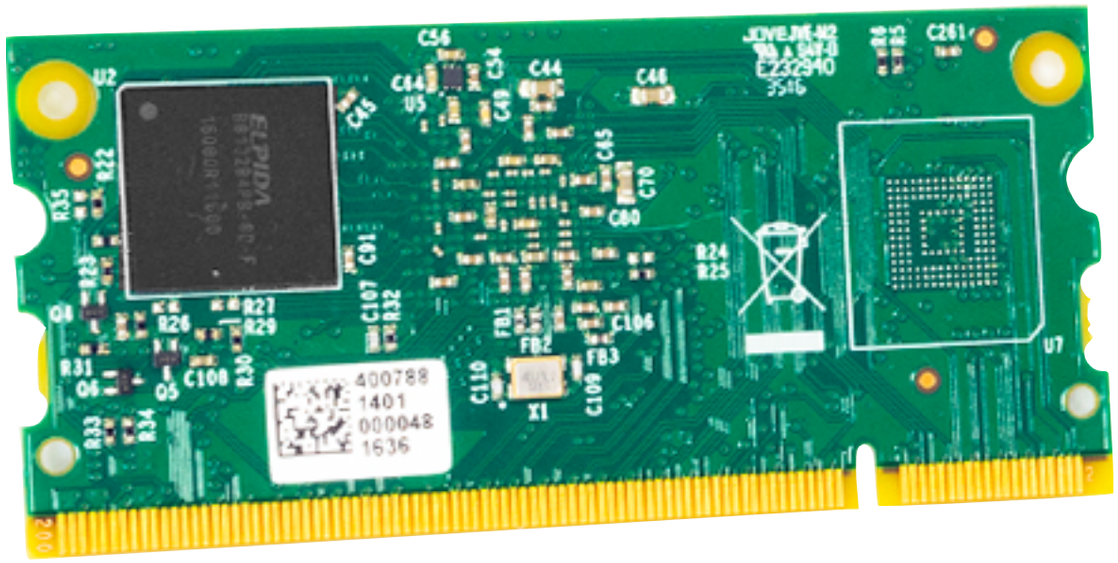
RaspberryPi Compute Module

Omega 2s



BeagleCore

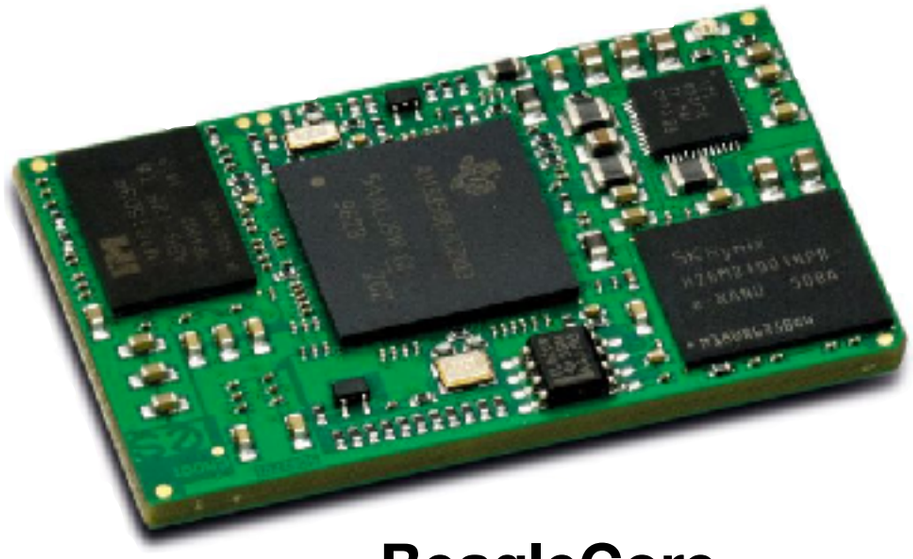
Chip Pro (not recommended)



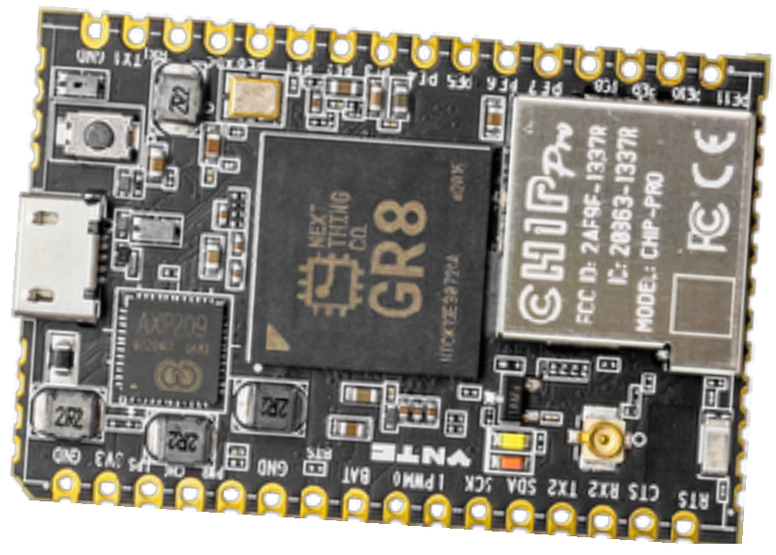
RaspberryPi Compute Module

Omega 2s

Many More…



BeagleCore

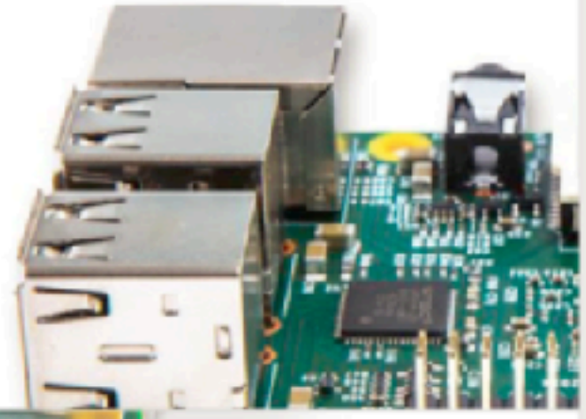


Chip Pro (not recommended)

Re-configure
the board layout

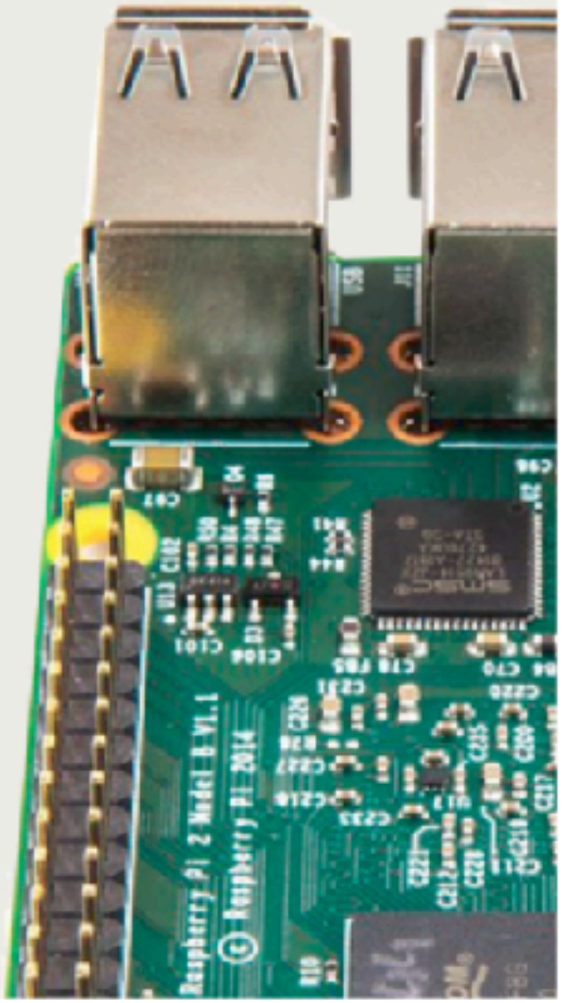
Incorporate
Additional
Functionality
onto the PCB

Add / Remove
Headers and Connectors



Add / Remove
Interfaces

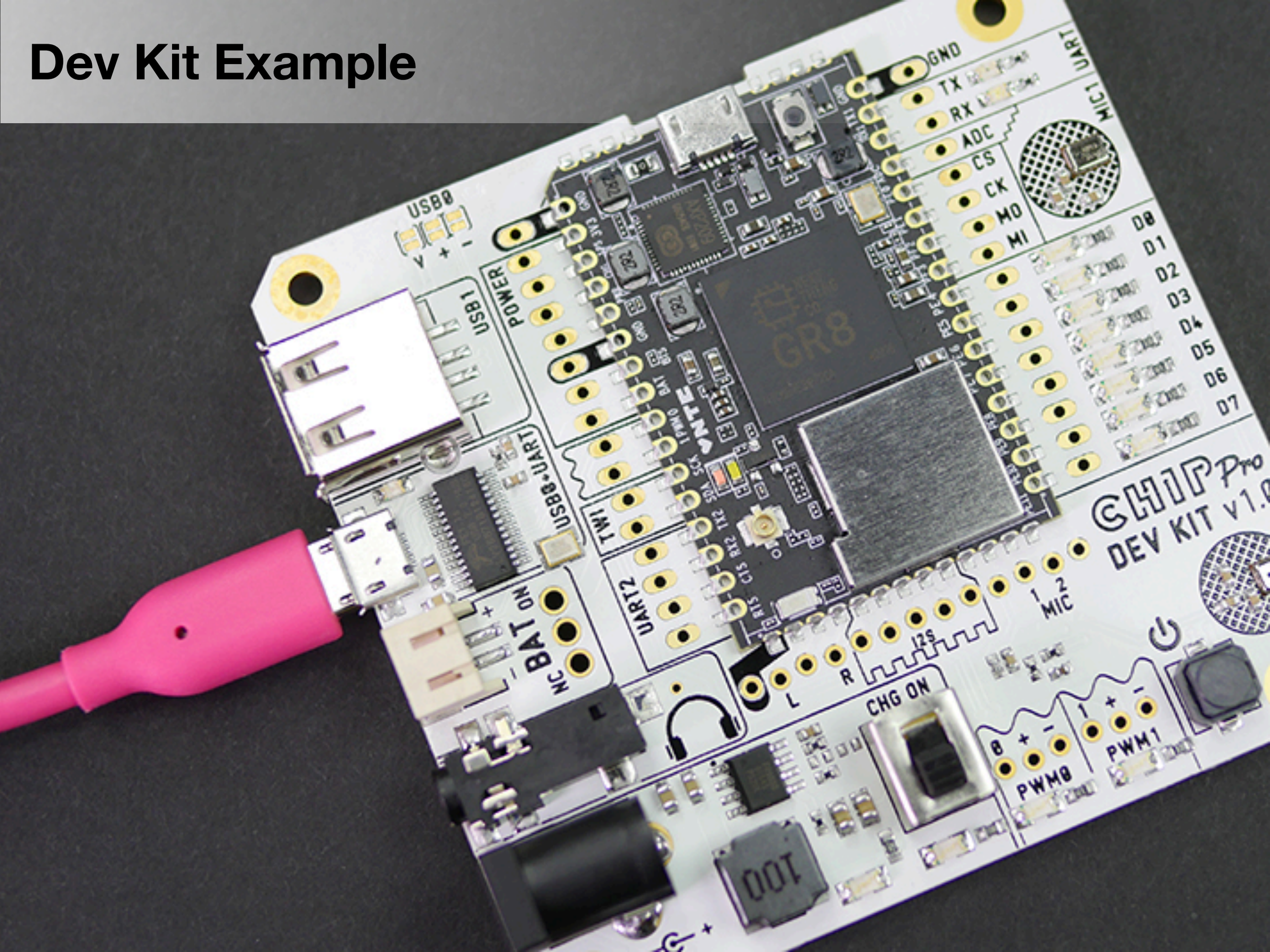
Change
Configuration



RaspberryPi Can Also Get Customized!

So How Are Those Things Programmed?

Dev Kit Example



And thankfully …

And thankfully …
Most Embedded Systems These Days Run…

And thankfully …
Most Embedded Systems These Days Run…

And thankfully …
Most Embedded Systems These Days Run…

This Includes…

ANDROID

Firefox OS

LG webOS

WIND™ AN INTEL COMPANY

… and others

```
    .data                   #data portion
strng:  .asciiz  " bottles of beer on the wall, "
strng2:  .asciiz  " bottles of beer "
strng3: .asciiz  "\ntake one down and pass it around, "
strng4: .asciiz  " bottle of beer on the wall "
strng5: .asciiz  " bottle of beer "
strng6: .asciiz  " bottles of beer on the wall.\n "
```

```
one: .word 1
     .text              #code section
main:                   #main
    li $a2, 1           #
    li $a3, 99          #start with 99
loop: jal PRNTB         #print bottle count
    la $a0, strng       #print strng
    li $v0, 4           #
    syscall             #
    jal PRNTB           #
    la $a0, strng2      #print strng2
    li $v0, 4           #
    syscall             #
    la $a0, strng3      #print strng3
    li $v0, 4           #
    syscall             #
    sub $a3, $a3, 1     #subtract one
    jal PRNTB           #
    la $a0, strng6      #print strng6
    li $v0, 4           #
    syscall             #
    bne $a3, $a2, skip  #handles one
                        #bottle on wall
    jal ONEBOT          #
skip: bnez $a3, loop    #Loop if not equal to 0
    li $v0, 10          #exit
    syscall             #
```

```
PRNTB: move $a0, $a3        #prnt bottle count
    li $v0, 1               #
    syscall                 #
    jr $ra                  #return from method

ONEBOT: addi $sp, $sp, -4   #allocate
    sw $ra, 0($sp)          #store rtrn address
    jal PRNTB               #
    la $a0, strng4          #print strng4
    li $v0, 4               #
    syscall                 #
    jal PRNTB               #
    la $a0, strng5          #print strng5
    li $v0, 4               #
    syscall                 #
    la $a0, strng3          #print strng3
    li $v0, 4               #
    syscall                 #
    sub $a3, $a3, 1         #subtract one
    jal PRNTB               #
    la $a0, strng6          #print strng6
    li $v0, 4               #
    syscall               #
    lw $ra,0($sp)           #load address
    addi $sp,$sp,4          #pop
    jr $ra                  #rtrn from method
```

Assembler

```
    .data                   #data portion
strng:  .asciiz  " bottles of beer on the wall, "
strng2:  .asciiz  " bottles of beer "
strng3: .asciiz  "\ntake one down and pass it around, "
strng4: .asciiz  " bottle of beer on the wall "
strng5: .asciiz  " bottle of beer "
strng6: .asciiz  " bottles of beer on the wall.\n "
```

```
one: .word 1
     .text              #code section
main:                   #main
    li $a2, 1           #
    li $a3, 99          #start with 99
loop: jal PRNTB         #print bottle count
    la $a0, strng       #print strng
    li $v0, 4           #
    syscall             #
    jal PRNTB           #
    la $a0, strng2      #print strng2
    li $v0, 4           #
    syscall             #
    la $a0, strng3      #print strng3
    li $v0, 4           #
    syscall             #
    sub $a3, $a3, 1     #subtract one
    jal PRNTB           #
    la $a0, strng6      #print strng6
    li $v0, 4           #
    syscall             #
    bne $a3, $a2, skip  #handles one
                        #bottle on wall
    jal ONEBOT          #
skip: bnez $a3, loop    #Loop if not equal to 0
    li $v0, 10          #exit
    syscall             #
```

```
PRNTB: move $a0, $a3        #prnt bottle count
    li $v0, 1               #
    syscall                 #
    jr $ra                  #return from method

ONEBOT: addi $sp, $sp, -4   #allocate
    sw $ra, 0($sp)          #store rtrn address
    jal PRNTB               #
    la $a0, strng4          #print strng4
    li $v0, 4               #
    syscall                 #
    jal PRNTB               #
    la $a0, strng5          #print strng5
    li $v0, 4               #
    syscall                 #
    la $a0, strng3          #print strng3
    li $v0, 4               #
    syscall                 #
    sub $a3, $a3, 1         #subtract one
    jal PRNTB               #
    la $a0, strng6          #print strng6
    li $v0, 4               #
    syscall              #
    lw $ra,0($sp)           #load address
    addi $sp,$sp,4          #pop
    jr $ra                  #rtrn from method
```

```
int main(void)
{
  int x;
  char xs[4];
  char *n = "o more";
  char *b = " bottle";
  char *o = " of beer";
  char *w = " on the wall";

  while(1)
  {
    x = 99;
    while(x)
    {
      printf("%d%s%s%s%s, ", x, b, x == 1 ? "" : "s", o, w);
      printf("%d%s%s%s\n", x, b, x == 1 ? "" : "s", o);
      printf("Take %s down and pass it around, ", x-- == 1 ? "it" : "one" );
      sprintf(xs, "%d", x);
      printf("%s%s%s%s%s\n\n", x > 0 ? xs : "No", b, x != 1 ? "s" : "", o, w );
    }
    printf("N%s%ss%s%s, n%s%ss%s\n\7", n, b, o, w, n, b, o);
    printf("Go to the store and buy some more\n");
  }
}
```

ANSI C

```
int main(void)
{
  int x;
  char xs[4];
  char *n = "o more";
  char *b = " bottle";
  char *o = " of beer";
  char *w = " on the wall";

  while(1)
  {
    x = 99;
    while(x)
    {
      printf("%d%s%s%s%s, ", x, b, x == 1 ? "" : "s", o, w);
      printf("%d%s%s%s\n", x, b, x == 1 ? "" : "s", o);
      printf("Take %s down and pass it around, ", x-- == 1 ? "it" : "one" );
      sprintf(xs, "%d", x);
      printf("%s%s%s%s%s\n\n", x > 0 ? xs : "No", b, x != 1 ? "s" : "", o, w );
    }
    printf("N%s%ss%s%s, n%s%ss%s\n\7", n, b, o, w, n, b, o);
    printf("Go to the store and buy some more\n");
  }
}
```

```
#include <iostream>
#include <string>

int main()
{
  std::string s[9] =
  {
    " bottle", " bottles", " of beer", " on the wall",
    "Take one down and pass it around, ", "No more", " no more", "1",
    "Go to the store and buy some more, 99"
  };
  for(int i = 99; i > 0; i—)
  {
    if (i < 3)
      std::cout << i << s[i - 1] << s[2] << s[3] << ", " << i << s[i- 1] << s[2]
      << ".\n" << s[4] << s[i + 5] << s[(i * -1) + 2] << s[2] << s[3] << ".\n\n";
    else
      std::cout << i << s[1] << s[2] << s[3] << ", " << i << s[1] << s[2]
      << ".\n" << s[4] << i - 1 << s[1] << s[2] << s[3] << “.\n\n";
  }
  std::cout << s[5] << s[1] << s[2] << s[3] << ',' << s[6] << s[1] << s[2]
  << ".\n" << s[8] << s[1] << s[2] << s[3] << '.';
  std::cin.get();
  return 0;
}
```

C++

```
#include <iostream>
#include <string>

int main()
{
  std::string s[9] =
  {
    " bottle", " bottles", " of beer", " on the wall",
    "Take one down and pass it around, ", "No more", " no more", "1",
    "Go to the store and buy some more, 99"
  };
  for(int i = 99; i > 0; i—)
  {
    if (i < 3)
      std::cout << i << s[i - 1] << s[2] << s[3] << ", " << i << s[i- 1] << s[2]
      << ".\n" << s[4] << s[i + 5] << s[(i * -1) + 2] << s[2] << s[3] << ".\n\n";
    else
      std::cout << i << s[1] << s[2] << s[3] << ", " << i << s[1] << s[2]
      << ".\n" << s[4] << i - 1 << s[1] << s[2] << s[3] << “.\n\n";
  }
  std::cout << s[5] << s[1] << s[2] << s[3] << ',' << s[6] << s[1] << s[2]
  << ".\n" << s[8] << s[1] << s[2] << s[3] << '.';
  std::cin.get();
  return 0;
}
```

… but I’m Not A Big Fan
Of Either Language

I’m A Java Guy!

I’m A Java Guy!

So The Obvious Choice Was?!

Oracle Java SE Embedded

Azul Zulu Embedded

Oracle Java SE Embedded

Azul Zulu Embedded

Space Requirement (Runtime)

Oracle Java SE Embedded

Azul Zulu Embedded

Space Requirement (Runtime)

Cost Requirement (Minimizing)

Couldn’t Afford Both

Java Out :(

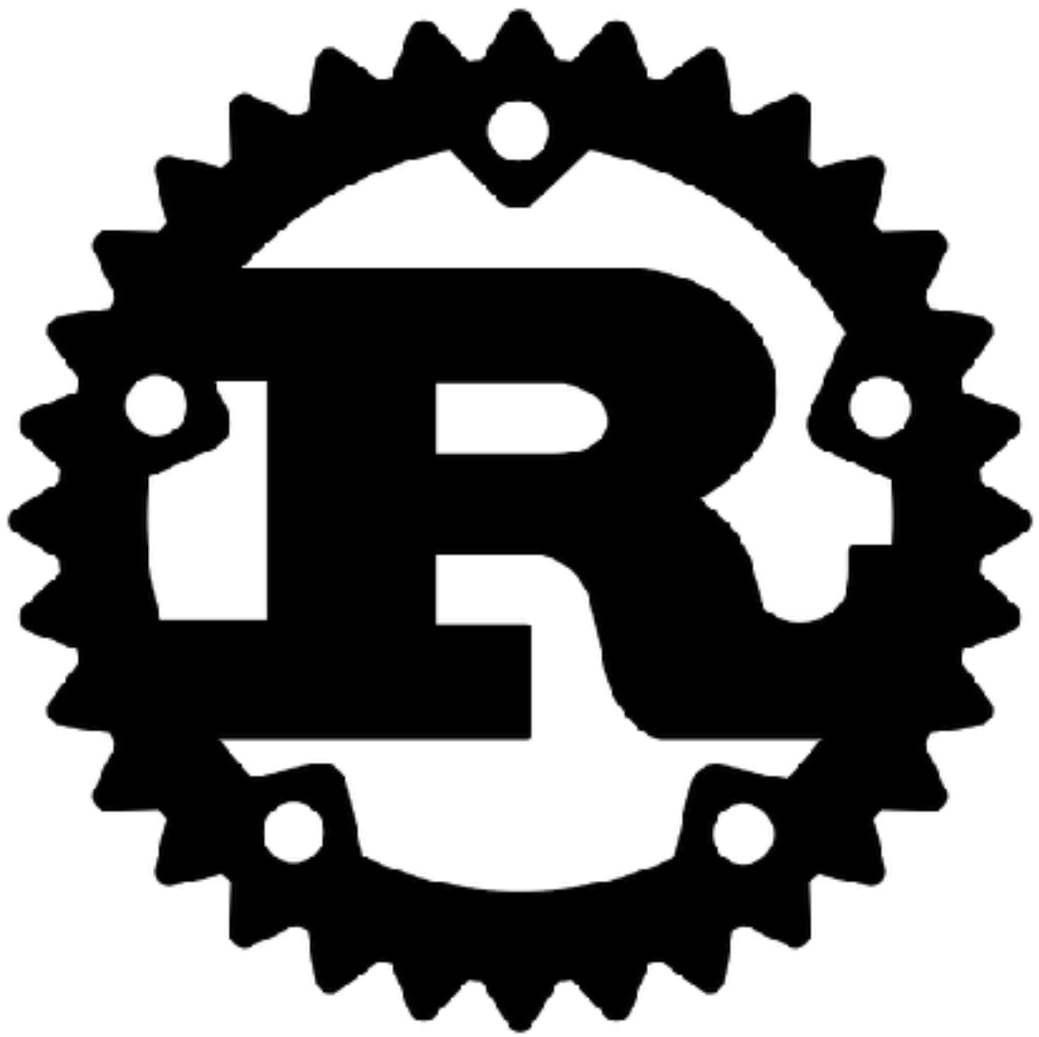
So I Went On A Journey…

- **I really don’t like C/C++**

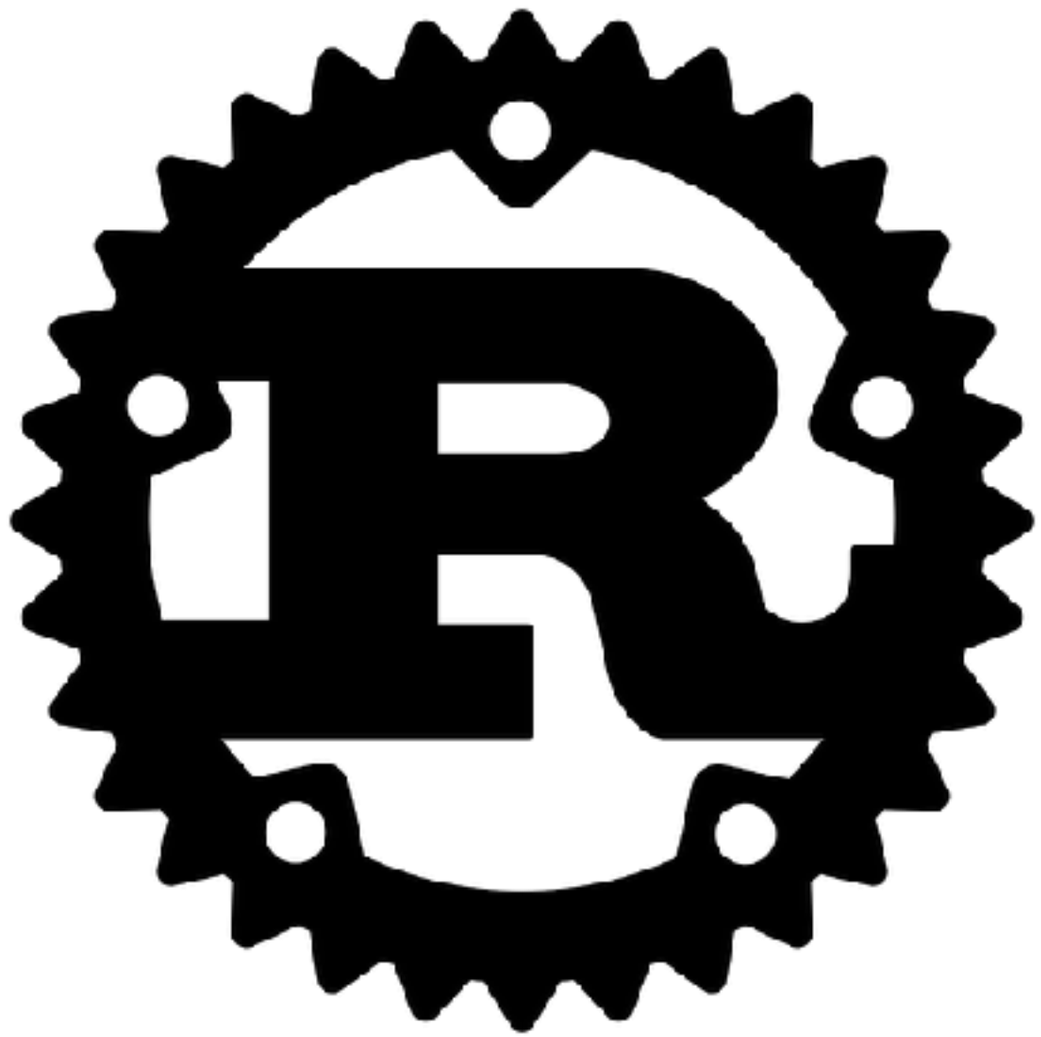
- **I really don’t like C/C++**
- **Complicated to cross-compile**

- **I really don’t like C/C++**
- **Complicated to cross-compile**
- **Hard to make secure / memory safe**

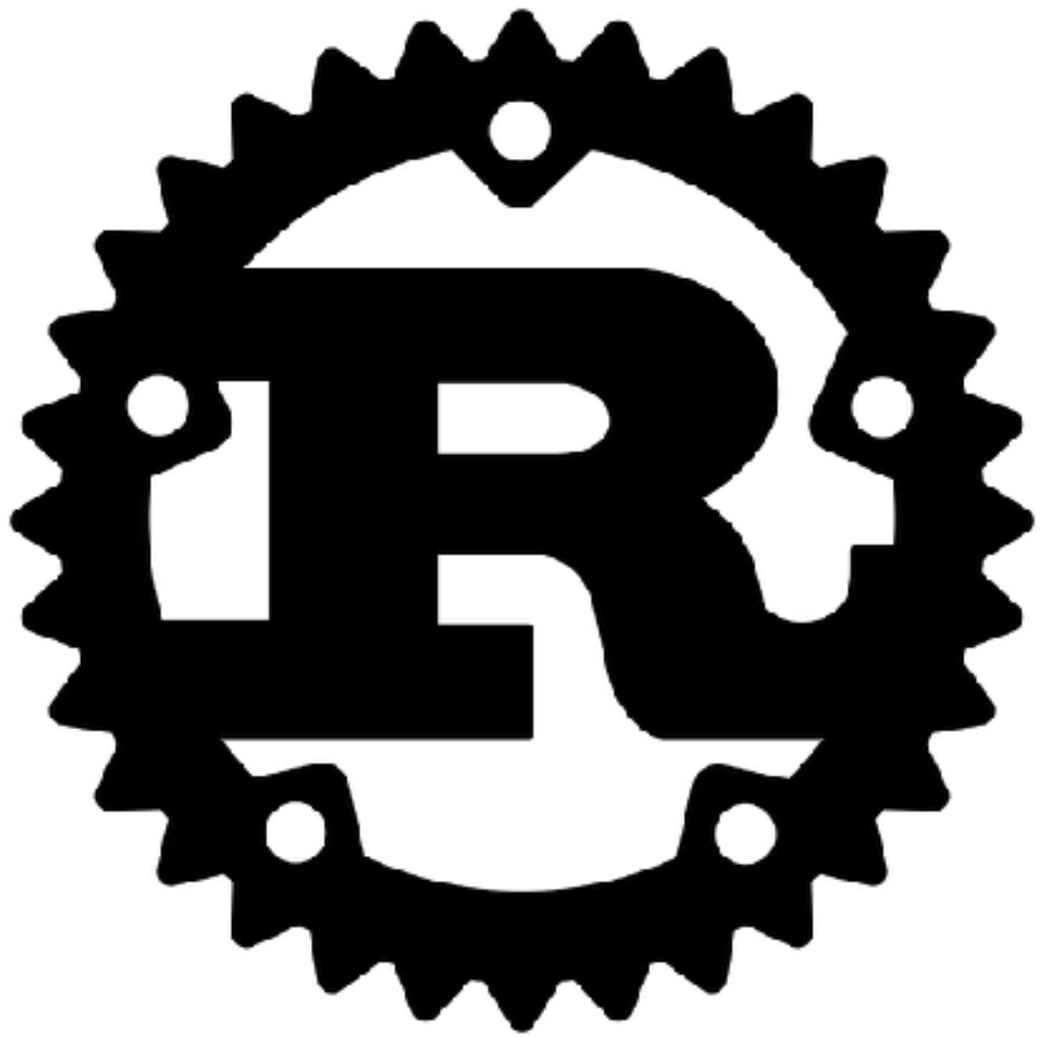
- **I really don’t like C/C++**
- **Complicated to cross-compile**
- **Hard to make secure / memory safe**
- **Custom memory handling**



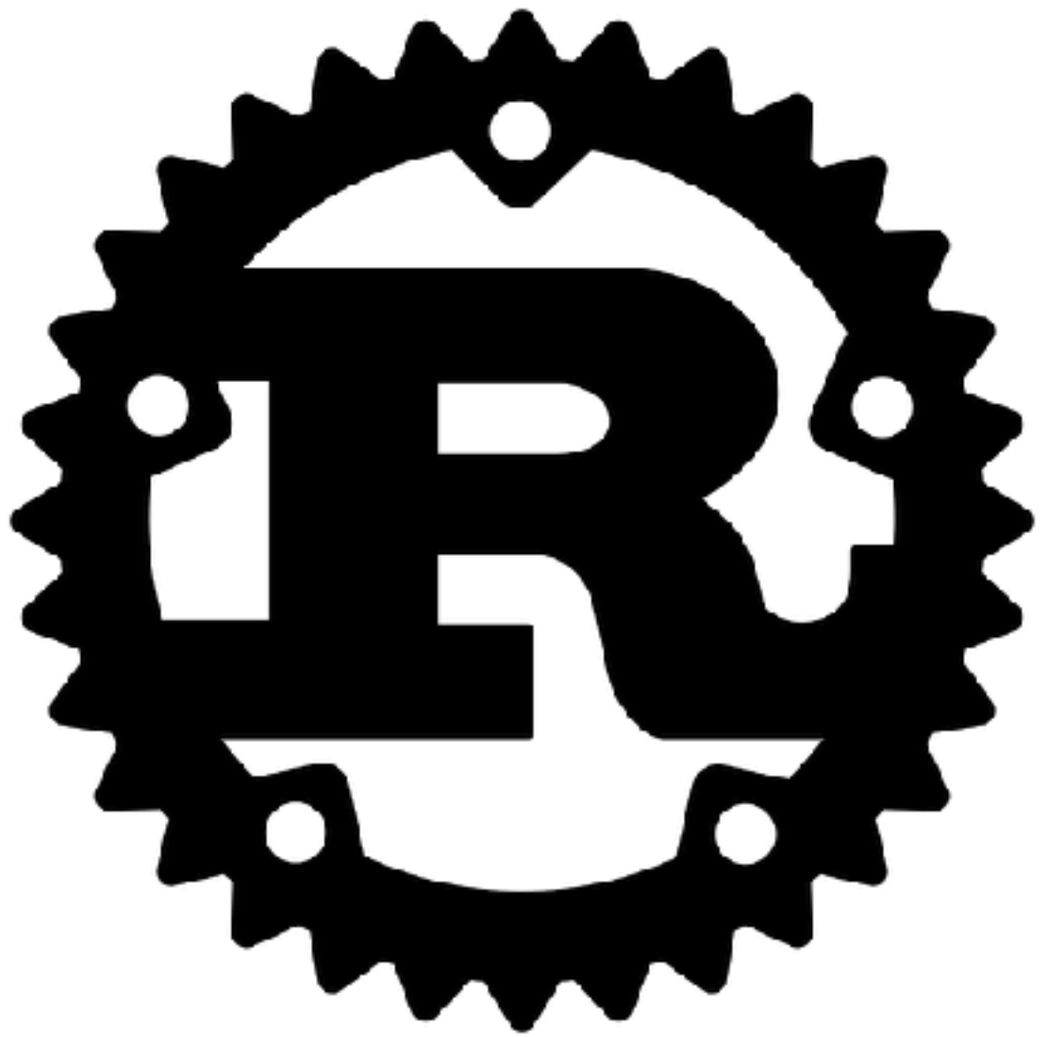
- **Interesting language**



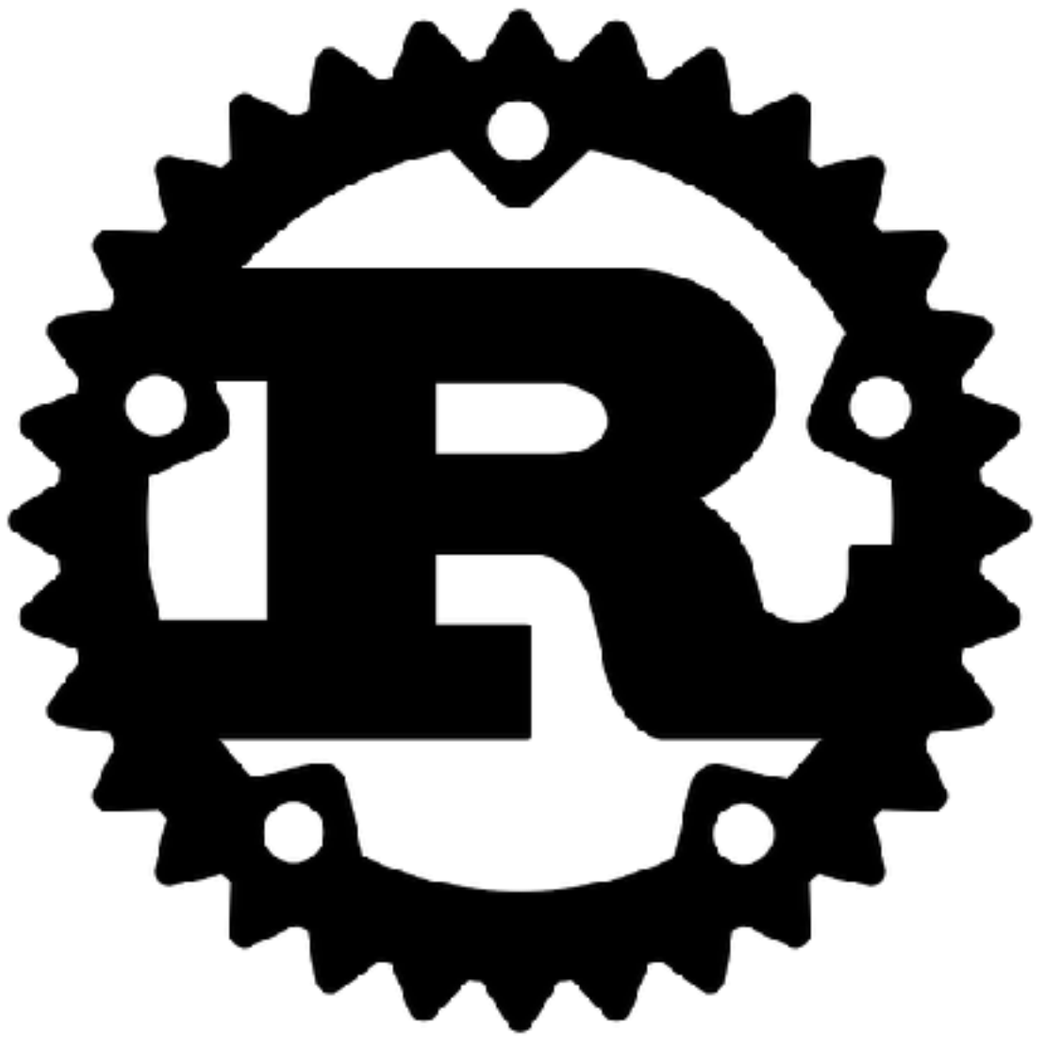
- **Interesting language**
- **Big community**



- **Interesting language**
- **Big community**
- **Very memory safe / borrowing**



- **Interesting language**
- **Big community**
- **Very memory safe / borrowing**
- **Still complex to cross-compile (cargo, rustup)**



Other Ideas?

Other Ideas?

Kotlin Native

Other Ideas?

Kotlin Native

Other Ideas?

Kotlin Native

Swift

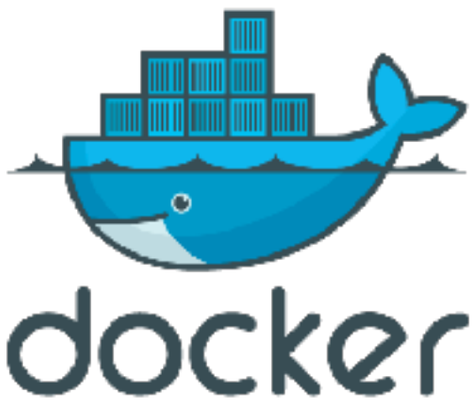

golang

- **Very opinionated language (I often like opinionated)**

- **Very opinionated language (I often like opinionated)**
- **Pretty big community**

- **Very opinionated language (I often like opinionated)**
- **Pretty big community, lots of known projects**

- **Very opinionated language (I often like opinionated)**
- **Pretty big community, lots of known projects**



- **Very opinionated language (I often like opinionated)**
- **Pretty big community, lots of known projects**

kubernetes

- **Very opinionated language (I often like opinionated)**
- **Pretty big community, lots of known projects**

etcd

kubernetes

- **Very opinionated language (I often like opinionated)**
- **Pretty big community, lots of known projects**
- **Very memory safe**

- **Very opinionated language (I often like opinionated)**
- **Pretty big community, lots of known projects**
- **Very memory safe**
- **Pretty cool cross-compile features (builtin)**

- **Very opinionated language (I often like opinionated)**
- **Pretty big community, lots of known projects**
- **Very memory safe**
- **Pretty cool cross-compile features (builtin)**
- **Kinda weird syntax**

- **Very opinionated language (I often like opinionated)**
- **Pretty big community, lots of known projects**
- **Very memory safe**
- **Pretty cool cross-compile features (builtin)**
- **Kinda weird syntax**
- **Error handling … don’t get me started ;-)**

EMBD.

Golang Embedded Programming Framework

CODE | DEMO | DOCS

Star Fork 134

http://embd.kidoman.io

ziutek / emgo

Watch 8 | Star 138 | Fork 12

Code | Issues 0 | Pull requests 0 | Projects 0 | Wiki | Insights

Emgo: Bare metal Go (language for programming embedded systems)

1,194 commits | 2 branches | 0 releases | 2 contributors | BSD-3-Clause

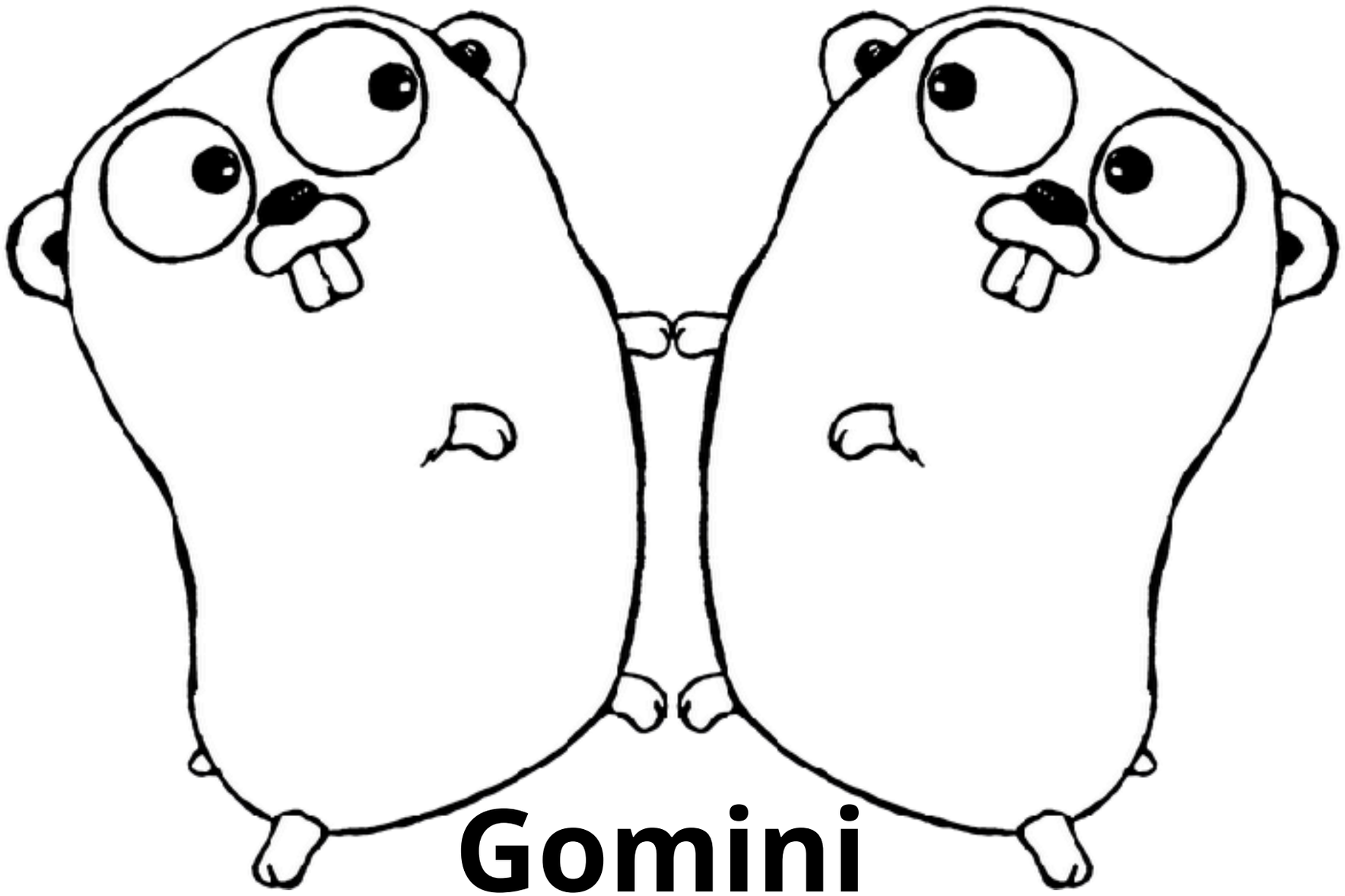
Branch: master | New pull request | Create new file | Upload files | Find file | Clone or download

ziutek examples/core51822: Introduce FanControl. Latest commit b41cdcf 9 days ago

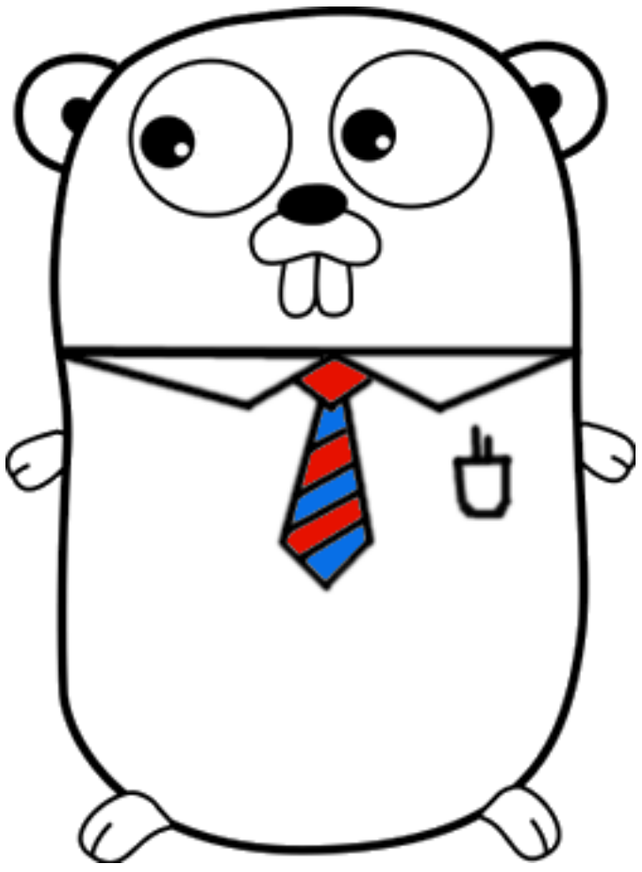
doc	doc: Fix code listings.	9 months ago
egc	egc: Add -Wno-discarded-qualifiers flag to allow slice array of mmio ...	3 months ago
egpath	examples/core51822: Introduce FanControl.	9 days ago
egroot	io: Add Closer, ReadCloser, WriteCloser, ReadWriteCloser.	2 months ago
gotoc-v1	Better support for stm32l1	4 years ago
gotoc	gotoc: Fix handling "init" names.	3 months ago
tools	stm32/hal: F303: Support for EXTI, update ADC code.	2 months ago
.gitignore	ld: Add ENTRY. Script names changed.	2 years ago
.gitmodules	math/matrix as submodule.	a year ago
LICENSE	Update LICENSE	10 months ago
README.md	examples: Improve README.	3 months ago
clean.sh	stm32/examples: Fixes...	a year ago
rebuild.sh	stm32/examples: Fixes...	a year ago

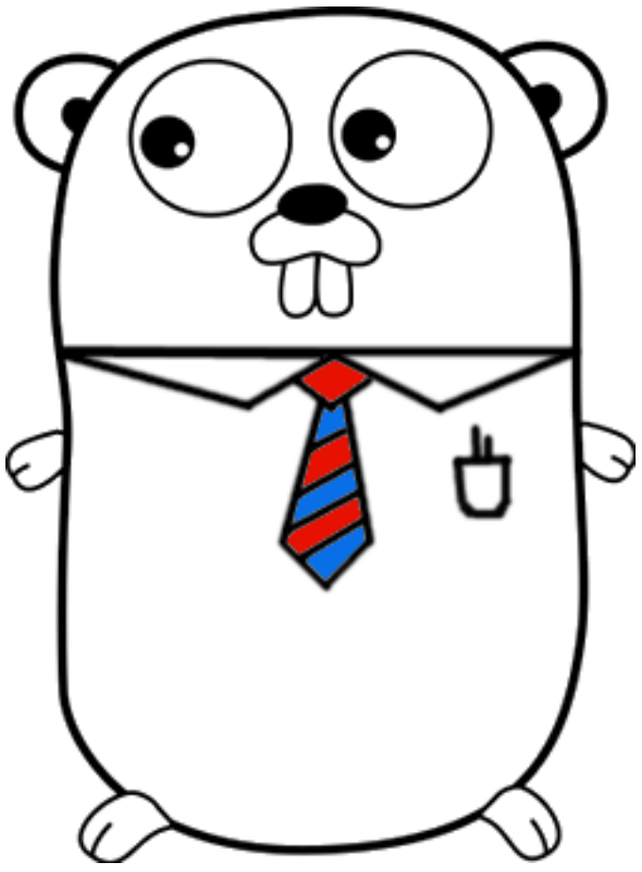
https://github.com/ziutek/emgo

Introducing



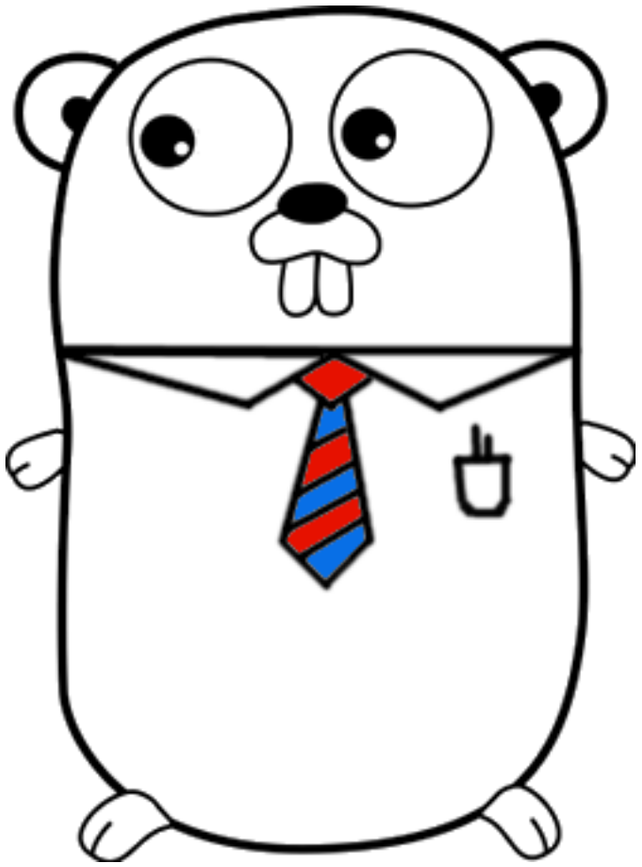
Gomini





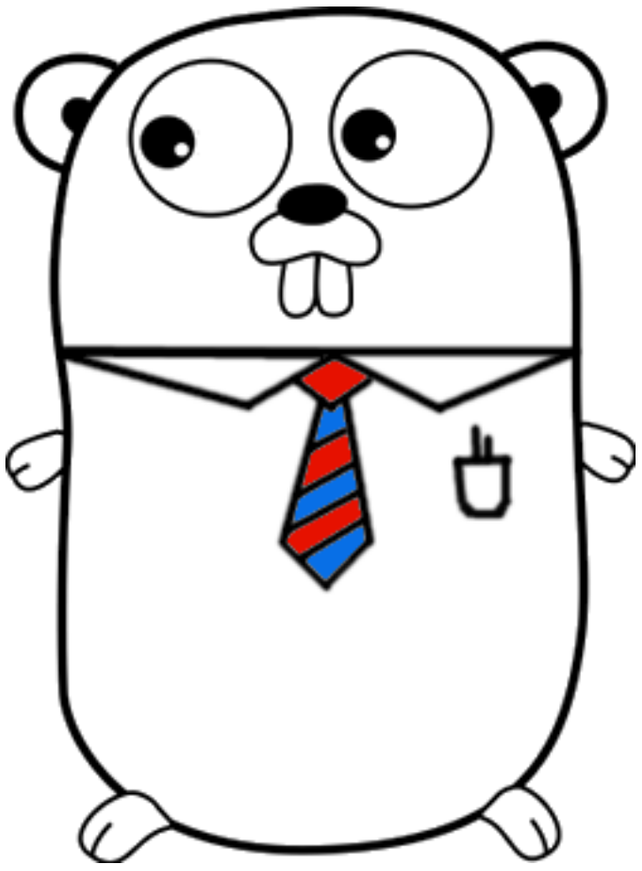
https://github.com/Shopify/go-lua

https://github.com/dop251/goja



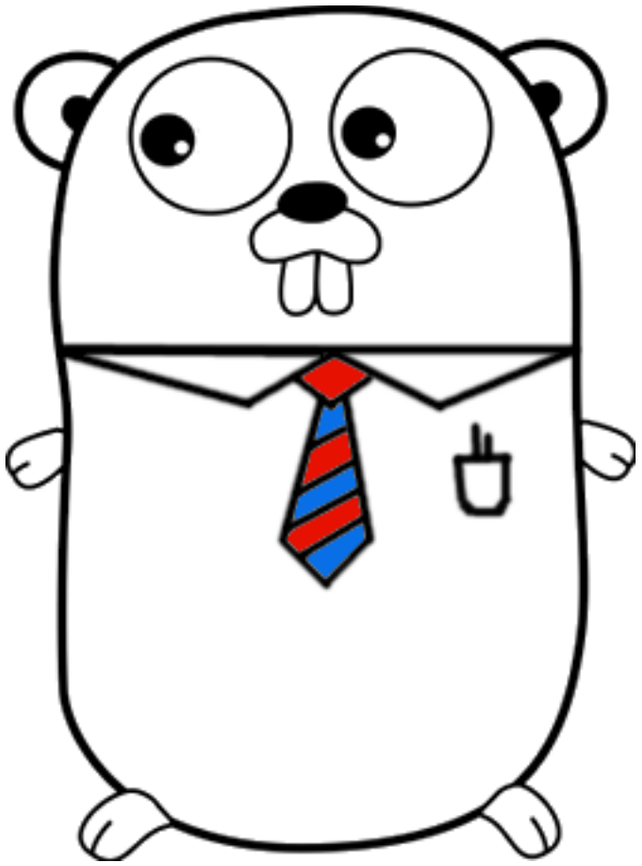
https://github.com/Shopify/go-lua

https://github.com/dop251/goja



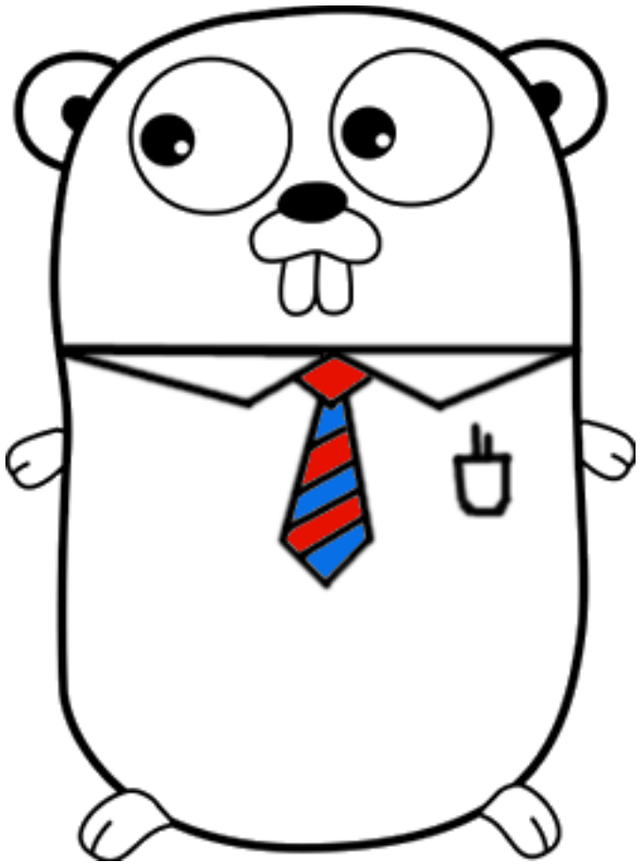
https://github.com/Shopify/go-lua

https://github.com/dop251/goja



https://github.com/Shopify/go-lua

https://github.com/dop251/goja



https://github.com/Shopify/go-lua

https://github.com/dop251/goja

So How Does Gomini Work?

Embedded Linux

Gomini Kernel (Golang)

Embedded Linux

Gomini Kernel (TypeScript)

Goja JS VM

Gomini Kernel (Golang)

Embedded Linux

Gomini Kernel (TypeScript)
Goja JS VM

App 1 (TypeScript)
Goja JS VM

App 2 (TypeScript)
Goja JS VM

...

App n (TypeScript)
Goja JS VM

Gomini Kernel (Golang)

Embedded Linux

Gomini Kernel (TypeScript)

Goja JS VM

ES6 JavaScript Proxy

App 1 (TypeScript)

Goja JS VM

App 2 (TypeScript)

Goja JS VM

...

App n (TypeScript)

Goja JS VM

Gomini Kernel (Golang)

Embedded Linux

Gomini Kernel

Gomini Kernel

- **JavaScript engine written in Go (dop251/goja)**

Gomini Kernel

- **JavaScript engine written in Go (dop251/goja)**
- **Kernel uses isolated JS VM**

Gomini Kernel

- JavaScript engine written in Go (dop251/goja)
- Kernel uses isolated JS VM
- Fully virtual filesystem

Gomini Kernel

- JavaScript engine written in Go (dop251/goja)
- Kernel uses isolated JS VM
- Fully virtual filesystem
  - `/kernel/*{.ts,.js}` (kernel code)

Gomini Kernel

- JavaScript engine written in Go (dop251/goja)
- Kernel uses isolated JS VM
- Fully virtual filesystem
  - `/kernel/*{.ts,.js}` (kernel code)
  - `/kernel/apps/*/*.ts` (deployed TypeScript apps)

Gomini Kernel

- JavaScript engine written in Go (dop251/goja)
- Kernel uses isolated JS VM
- Fully virtual filesystem
  - `/kernel/*{.ts,.js}` (kernel code)
  - `/kernel/apps/*/*.ts` (deployed TypeScript apps)
  - `/kernel/@types` (exported kernel APIs .d.ts files)

Gomini Kernel

- JavaScript engine written in Go (dop251/goja)
- Kernel uses isolated JS VM
- Fully virtual filesystem
  - `/kernel/*{.ts,.js}` (kernel code)
  - `/kernel/apps/*/*.ts` (deployed TypeScript apps)
  - `/kernel/@types` (exported kernel APIs .d.ts files)
  - `/kernel/cache` (transpiler cache)

Gomini Apps

Gomini Apps

- **Each uses an isolated JS VM**

Gomini Apps

- **Each uses an isolated JS VM**
- **Access to kernel APIs using ES6 Proxies**

Gomini Apps

- Each uses an isolated JS VM
- Access to kernel APIs using ES6 Proxies
- Again fully virtual filesystem

Gomini Apps

- Each uses an isolated JS VM
- Access to kernel APIs using ES6 Proxies
- Again fully virtual filesystem
  - `/` (in kernel `/kernel/apps/appx/`)

Gomini Apps

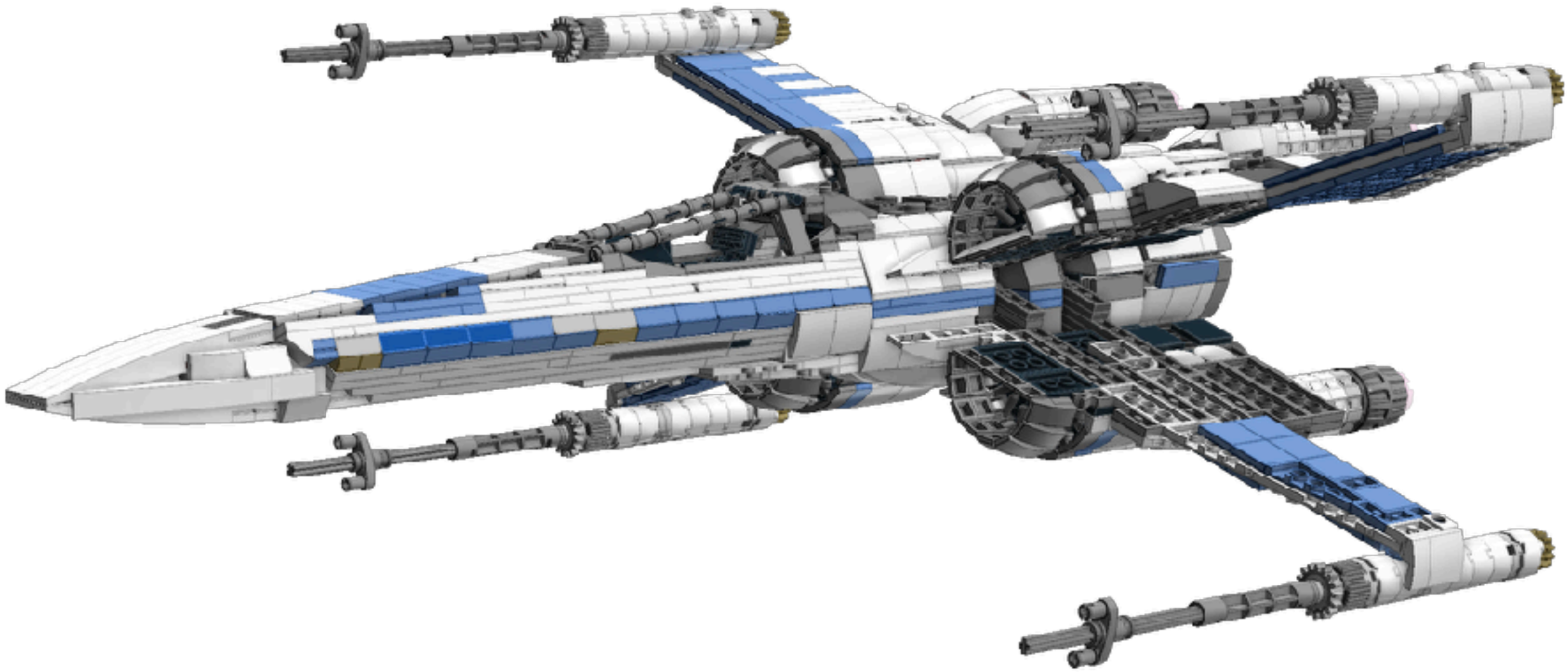
- Each uses an isolated JS VM
- Access to kernel APIs using ES6 Proxies
- Again fully virtual filesystem
  - `/` (in kernel `/kernel/apps/appx/`)
  - `/*/*{.ts,.js}` (app code)

Gomini Apps

- Each uses an isolated JS VM
- Access to kernel APIs using ES6 Proxies
- Again fully virtual filesystem
  - `/` (in kernel `/kernel/apps/appx/`)
  - `/*/*{.ts,.js}` (app code)
  - `/kernel/@types` (exported kernel APIs .d.ts files)

Gomini Apps

- Each uses an isolated JS VM
- Access to kernel APIs using ES6 Proxies
- Again fully virtual filesystem
  - `/` (in kernel `/kernel/apps/appx/`)
  - `/*/*{.ts,.js}` (app code)
  - `/kernel/@types` (exported kernel APIs .d.ts files)
  - other directories based on permissions

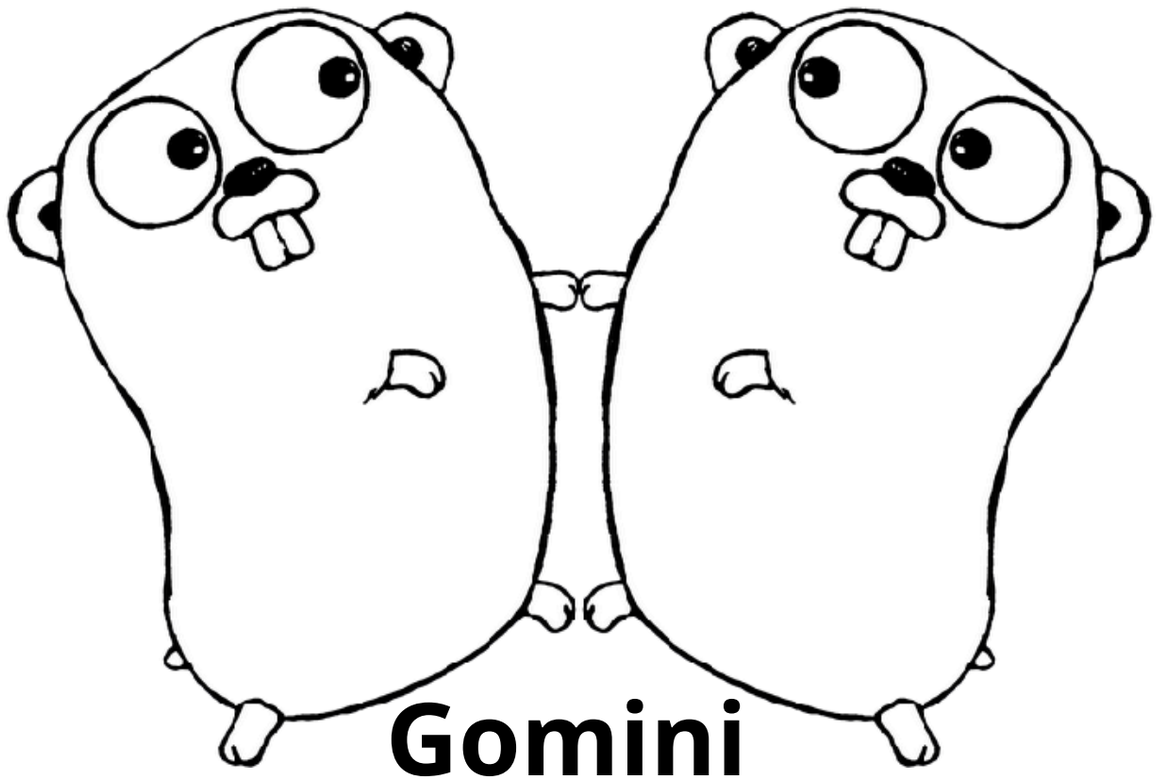


Ok Enough Theory, Time To Play

Thank You
Questions?

https://github.com/noctarius/gomini-example

https://github.com/relationsone/gomini



Gomini

Thank You
Questions?

https://github.com/noctarius/gomini-example

https://github.com/relationsone/gomini

https://www.kickstarter.com/projects/1598272670/meet-otto-the-hackable-gif-camera
https://software.intel.com/en-us/iot/hardware/discontinued
https://kotlinlang.org/docs/reference/native-overview.html
https://www.oracle.com/java/java-se-embedded.html
https://www.azul.com/products/zulu-embedded
http://www.ingenic.com/en/?newton/id/13.html
https://www.raspberrypi.org
http://beaglecore.com
http://beagleboard.org
https://getchip.com/pages/chippro
http://www.orangepi.org
https://pi-top.com
http://fiveninjas.com
http://dock2office.com
https://onion.io
http://www.element14.com/custompi
https://github.com/rust-lang/cargo
https://www.rustup.rs
https://nodejs.org/en
https://developer.apple.com/swift
http://embd.kidoman.io
https://github.com/ziutek/emgo
https://github.com/Shopify/go-lua
https://github.com/dop251/goja