

The Internet of Programmable Things

Kasper Lund

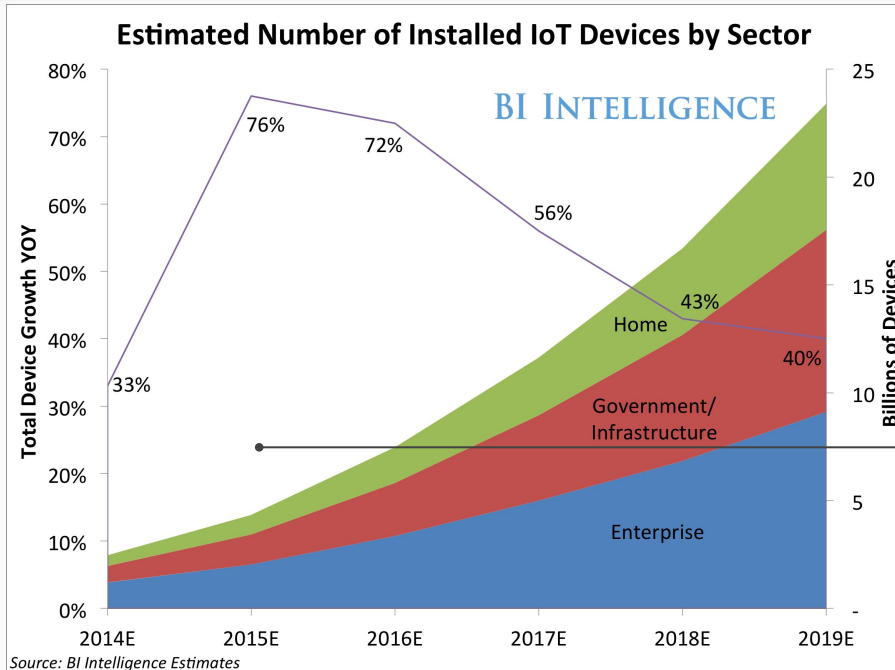
Background

Kasper Lund, software engineer @ Google

Projects:

- **OOVM** embedded Smalltalk system
- **V8** high-performance JavaScript engine
- **Dart** structured programming for the web

An explosion of devices



number of cellphones today

IoT: Internet of things

- **sense:** data is gathered, processed and transmitted
- **transport:** data passes over various networks
- **store:** information is gathered and stored
- **analyze:** insights are extracted and presented
- **control:** based on insights alerts are sent or devices take action
- **share:** data is exchanged with other systems

Devices for sense and control



microcontroller (MCU)



microprocessor (MPU)

Prototypical microprocessor

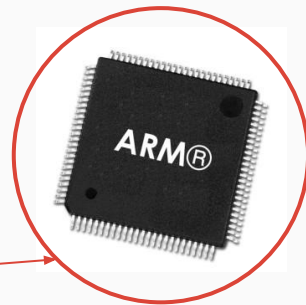
- **Raspberry Pi 2: Model B**
 - ARMv7 Quad Core Processor @ 900 MHz
 - 1 GB of RAM
 - Plenty of Flash through external MicroSD-card
 - **Price:** ~45 USD
- Runs a variety of operating systems
 - Linux (e.g. Raspbian), RISC OS, FreeBSD, NetBSD, Plan 9, Inferno, AROS
 - Even Windows 10 IoT Core



Prototypical microcontroller

- **STM32F7:** high-performance MCU with ARM Cortex-M7 core
 - 32-bit ARM CPU @ 216 MHz
 - 320 KB of RAM
 - 512-1024 KB of Flash
 - Single precision FPU and no fancy MMU
 - **Price:** ~12 USD in low quantity bulk

our focus



I Has a Bucket

can i haz
embedded
linux?

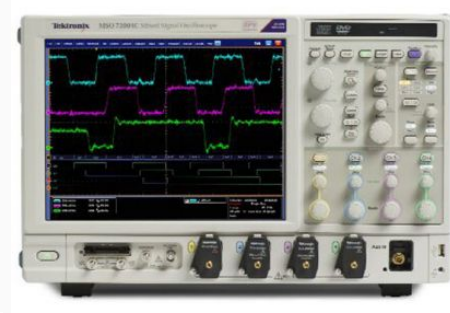
Noooo they ha

you mean
uClinux?

do you
4MB
AM?
2-3MB of
Flash?



What are we left with?



Best case scenario

```
1 #include "mbed.h"
2 #include "MMA8451Q.h"
3
4 #define MMA8451_I2C_ADDRESS (0x1d<<1)
5
6 int main(void) {
7     MMA8451Q acc(PTE25, PTE24, MMA8451_I2C_ADDRESS);
8     PwmOut rled(LED_RED);
9     PwmOut gled(LED_GREEN);
10    PwmOut bled(LED_BLUE);
11
12    while (true) {
13        rled = 1.0 - abs(acc.getAccX());
14        gled = 1.0 - abs(acc.getAccY());
15        bled = 1.0 - abs(acc.getAccZ());
16        wait(0.1);
17    }
18 }
```

```
$ yt debug example-mbedos-blinky
info: found example-mbedos-blinky at source/example-mbedos-blinky
info: starting PyOCD gdbserver...
info: new board id detected: 02400201C37A4E793E84B3C1
info: board allows 5 concurrent packets
info: DAP SWD MODE initialised
info: IDCODE: 0x2BA01477
info: K64F not in secure state
info: 6 hardware breakpoints, 4 literal comparators
info: CPU core is Cortex-M4
info: FPU present
info: 4 hardware watchpoints
info: Telnet: server started on port 4444
info: GDB server started at port:3333
GNU gdb (GNU Tools for ARM Embedded Processors) 7.6.0.20140731-cvs
Copyright (C) 2013 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law. Type "show copying"
and "show warranty" for details.
This GDB was configured as "--host=x86_64-apple-darwin10 --target=arm-none-eabi".
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>...
Reading symbols
from
/examples/example-mbedos-blinky/build/frdm-k64f-gcc/source/example-mbedos-blinky
...done.
info: One client connected!
(gdb)
```

Let me tell you what I really want

Let me tell you what you really need

You need an accessible platform

... that seems instantly familiar to the masses

... based on a high-level, managed language

... to make devices programmable by everyone

You need a productive environment

... with solid support for static analysis and code completion

... and a rich ecosystem of support functionality in libraries

... to allow developers to build great things quickly

We all need an open, serviceable platform

... that provides safety guarantees for upgradable components

... that allows independently developed code to co-exist

... in a new era of truly extensible and configurable devices

So all we

... a modern,

... with tools

... and librari

... that supp

... and all of i



gation

Is

anged

Dart

```
void main() {  
  var greetings = "Hello, World";  
  print(greetings);  
}
```

try for yourself by visiting dartpad.dartlang.org



MCU +  ?

does Dart run on microcontrollers at all?

Sneak peek of the Fletch project

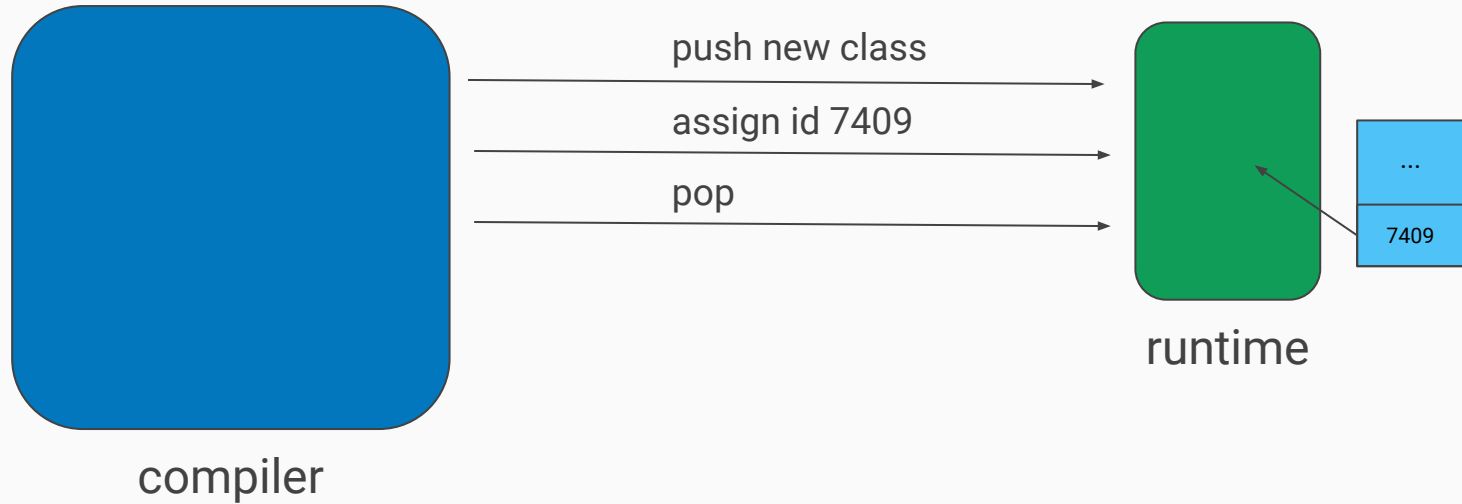
Characteristics

- small and light
- productive and serviceable
- simple and accessible

What takes space in a virtual machine?

- source-code compiler
- execution engine
- object model
- garbage collector
- debugging support

Interface between compiler and runtime



Reflection over wire protocol

- set up the program structure
 - push new class
 - push new method
- change the program
 - change method
 - change schema
- debug the runtime
 - set breakpoint
 - restart activation

The runtime is a **small**, stack-based machine driven from the outside by the compiler.

Optimizing dynamic dispatching

(super short interlude)

Dynamic dispatching

```
class A {  
    bar();  
}
```

```
class B extends A {  
    foo();  
}
```

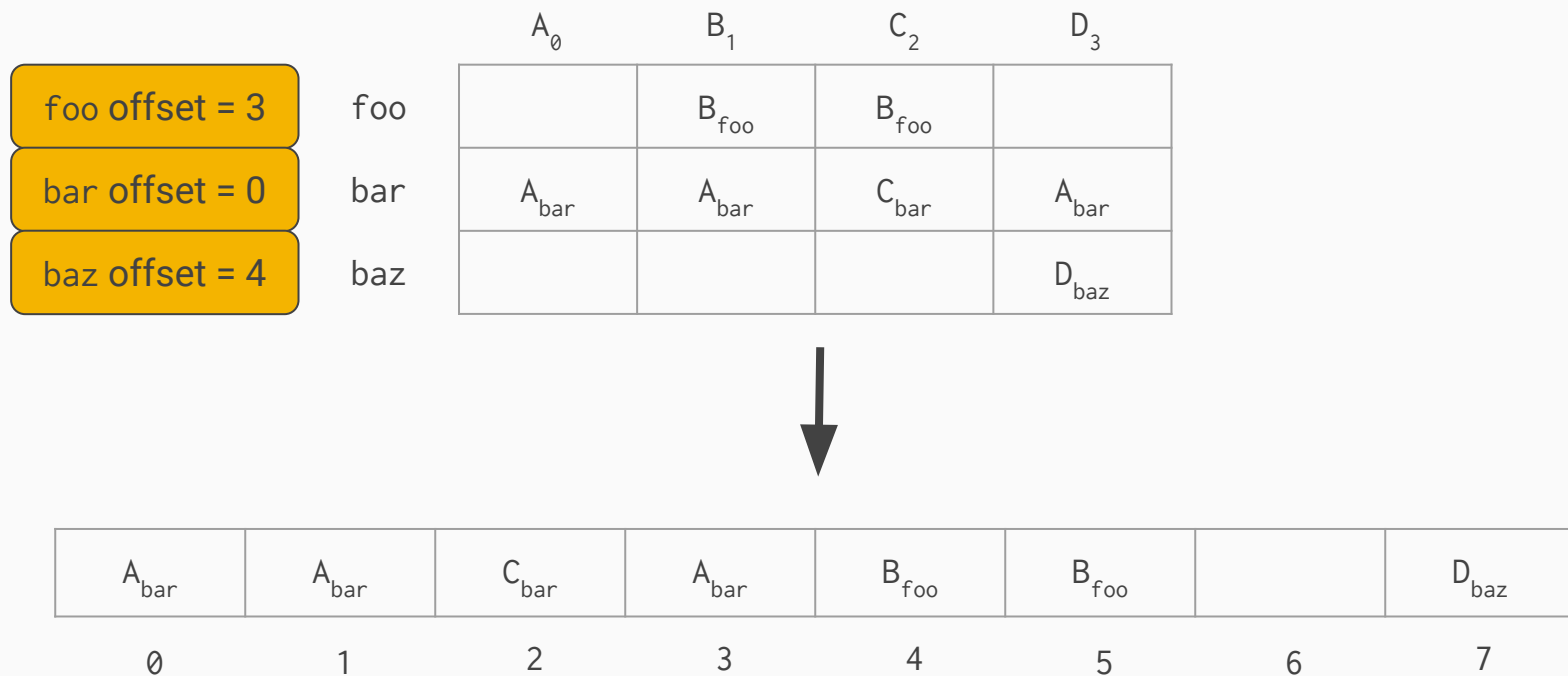
```
class C extends B {  
    bar();  
}
```

```
class D extends A {  
    baz();  
}
```



	A	B	C	D
foo		B _{foo}	B _{foo}	
bar	A _{bar}	A _{bar}	C _{bar}	A _{bar}
baz				D _{baz}

Selector-based row displacement

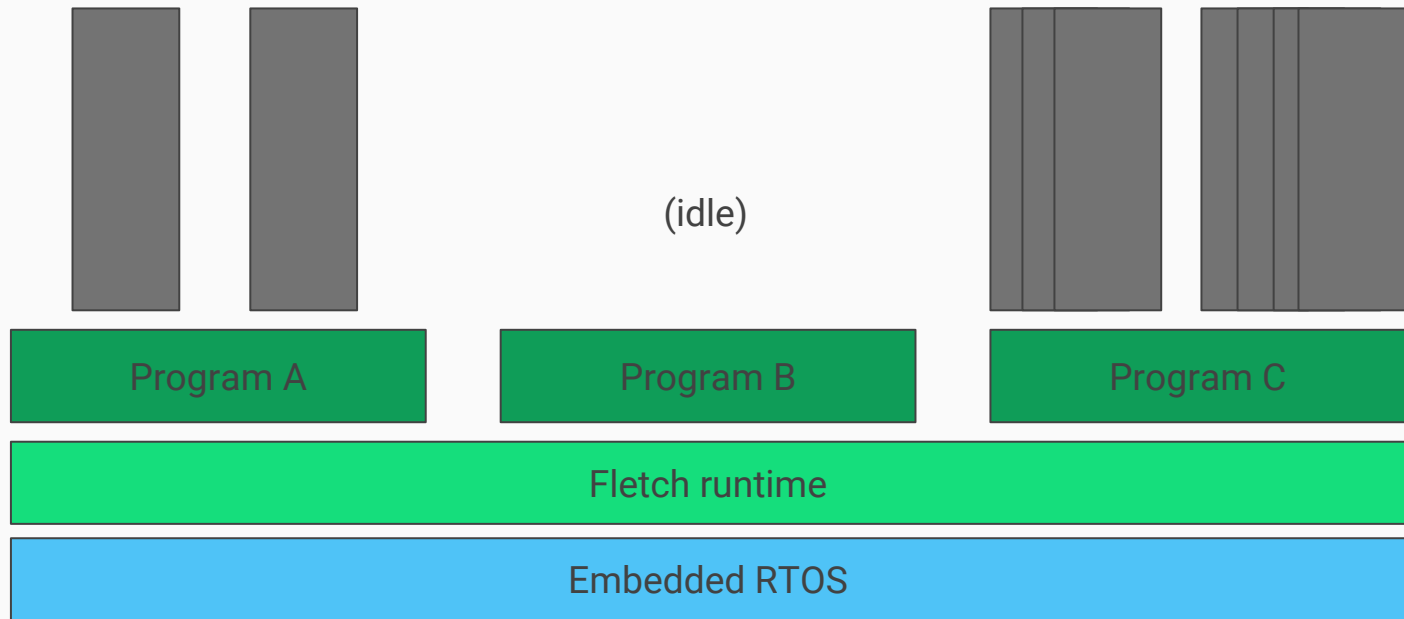


Selector-based row displacement

- guaranteed constant time dynamic dispatching
- table is precomputed and part of the application snapshot
- takes up 10-17% more space than vtables

Programming model

Programs and processes



Light-weight processes

- small memory footprint (~4K)
- can be blocked without taking up system resources
- run in parallel if you have enough cores

Handle connections in new processes

```
var server = new ServerSocket("127.0.0.1", 0);
while (true) {
    server.spawnAccept((Socket socket) {
        // Runs in a new process.
        socket.send(UTF8.encode("Hello, World"));
        socket.close();
    });
}
```


Blocking messaging

```
final channel = new Channel();    // A channel is a queue of messages.  
final port = new Port(channel);   // A port is the capability to send to a channel.
```

```
Process.spawn(() {  
    int i = 0;  
    while (true) port.send(i++);  
});
```

```
while (true) print(channel.receive());
```

too easy?

the only messages sent between processes are integers...

Shared state concurrency

- any object can be sent as a message without copying
- multiple processes can operate on the objects in parallel
- lots of explicit synchronization primitives

Shared immutable state concurrency

- any deeply immutable object can be sent as a message without copying
- multiple processes can operate on the objects in parallel
- no need for explicit synchronization primitives

one more thing

can a process wait for more than one thing?

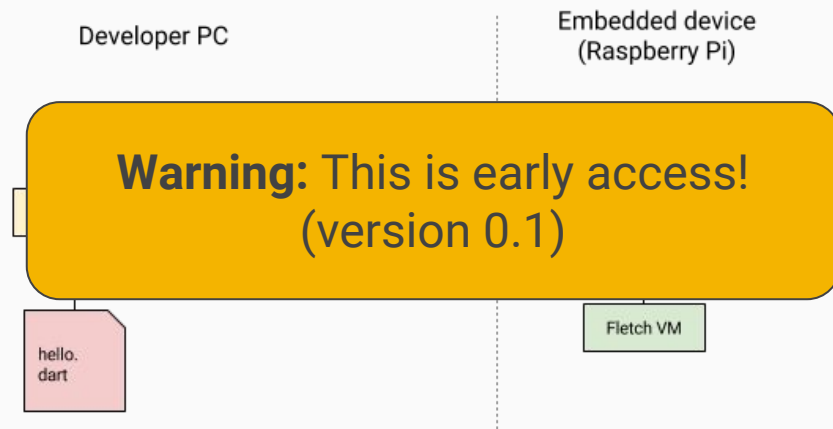
Fibers!

```
void publishOnChange(Socket socket, String propertyName, Channel input) {  
    int last = 0;  
    while (true) {  
        int current = input.receive();  
        if (current != last) socket.send(UTF8.encode('{ "$propertyName": $current }'));  
        last = current;  
    }  
}
```

```
Fiber.fork(() => publishOnChange(server, "temperature", temperatureSensor));  
Fiber.fork(() => publishOnChange(server, "humidity", humiditySensor));
```

All that on an embedded device...

Fletch SDK for Raspberry Pi 2



<http://dart-lang.github.io/fletch/>

Running code

```
$ fletch run samples/general/hello.dart
```

```
Hello from Darwin running on kasperl-macbookpro.
```

```
$ fletch run samples/general/hello.dart in session remote
```

```
Hello from Linux running on raspberrypi.
```

Blinking lights on the Raspberry Pi 2

```
import 'package:gpio/gpio.dart';  
import 'package:os/os.dart';  
  
const int PI_ONBOARD_GREEN_LED = 47;  
  
main() {  
  var gpio = new PiMemoryMappedGPIO();  
  gpio.setMode(PI_ONBOARD_GREEN_LED, Mode.output);  
  for (bool enabled = true; true; enabled = !enabled) {  
    gpio.setPin(PI_ONBOARD_GREEN_LED, enabled);  
    sleep(500);  
  }  
}
```

TL;DR

The world needs more productive
embedded developers

Too many developers lack the
skills to target embedded devices

Productivity is higher with great tools and managed languages

Managed languages that fit on
microcontrollers soon include Dart

Which device are you going to
hack on next?



**Click 'engage'
to rate sessions
and ask questions**



Please

**Remember to
rate this session**

Thank you!

