

Making the Ada Drivers Library

Embedded Programming with Ada

Fabien Chouteau

Embedded Software Engineer at AdaCore

 Twitter : @DesChips

 GitHub : Fabien-Chouteau

 Hackaday.io: Fabien.C

**Programming is all about
communication**

Programming is all about communication

With:

- The compiler

Programming is all about communication

With:

- The compiler
- The other tools (static analyzers, provers, etc.)

Programming is all about communication

With:

- The compiler
- The other tools (static analyzers, provers, etc.)
- Users of your API

Programming is all about communication

With:

- The compiler
- The other tools (static analyzers, provers, etc.)
- Users of your API
- Your colleagues

Programming is all about communication

With:

- The compiler
- The other tools (static analyzers, provers, etc.)
- Users of your API
- Your colleagues
- The idiot that wrote this stupid piece of code...

Programming is all about communication

With:

- The compiler
- The other tools (static analyzers, provers, etc.)
- Users of your API
- Your colleagues
- The idiot that wrote this stupid piece of code...
- **Oh, wait. It was me two months ago :(**

What makes Embedded Programming different?

Every bug costs more:

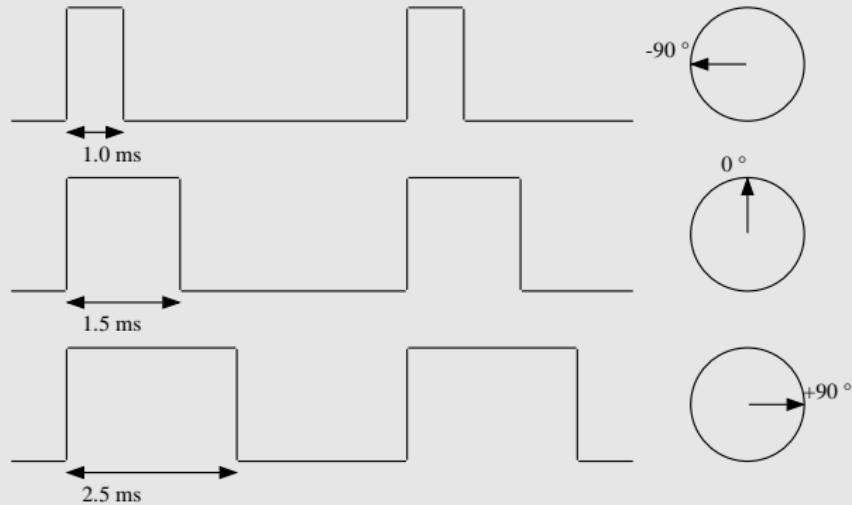
- More time to investigate
- More time to try a fix
- Potential destruction of hardware
- Updates are difficult

You need more control:

- Low resources (RAM, flash, CPU)
- Interaction with the hardware
- Real-Time constraints

Embedded Programming with Ada

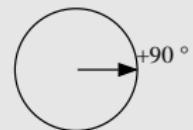
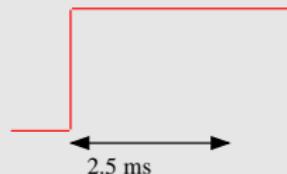
Servo motor example



Servo motor example



Servo motor example



Types

```
procedure Set_Angle (Angle : Integer);
```

Types

```
-- Set desired angle for the servo motor
--
-- @param Angle: Desired rotation angle in degree.
-- Please do not use a value above 90 or below -90!
procedure Set_Angle (Angle : Integer);
```

Types

```
type Servo_Angle is range -90 .. 90;  
--  Servo rotation angle in degree  
  
procedure Set_Angle (Angle : Servo_Angle);  
--  Set desired angle for the servo motor
```

The compiler: GNAT

```
Set_Angle (100);
```

```
warning: value not in range of type "Servo_Angle"  
warning: "Constraint_Error" will be raised at run time
```

The static analyzer: CodePeer

```
procedure Set_Angle_Double (X : Servo_Angle) is
begin
    Set_Angle (X * 2);
end Set_Angle_Double;

Set_Angle_Double (80);
```

servo_driver.adb:27:4: high: precondition (range check) failure on
call to servo_driver.set_angle_double: requires X in -45..45

The formal proof: SPARK

```
Phase 1 of 2: generation of Global contracts ...
servo_driver.adb:42:04: error in inlined body at line 23
servo_driver.adb:42:04: value not in range of type
    "Servo_Angle" defined at line 7
servo_driver.adb:42:04: "Constraint_Error" would have
    been raised at run time
```

The debugger: Gdb

```
(gdb) catch exception
Catchpoint 1: all Ada exceptions
(gdb) run

Catchpoint 1, CONSTRAINT_ERROR
(servo_driver.adb:23 overflow check failed)
```

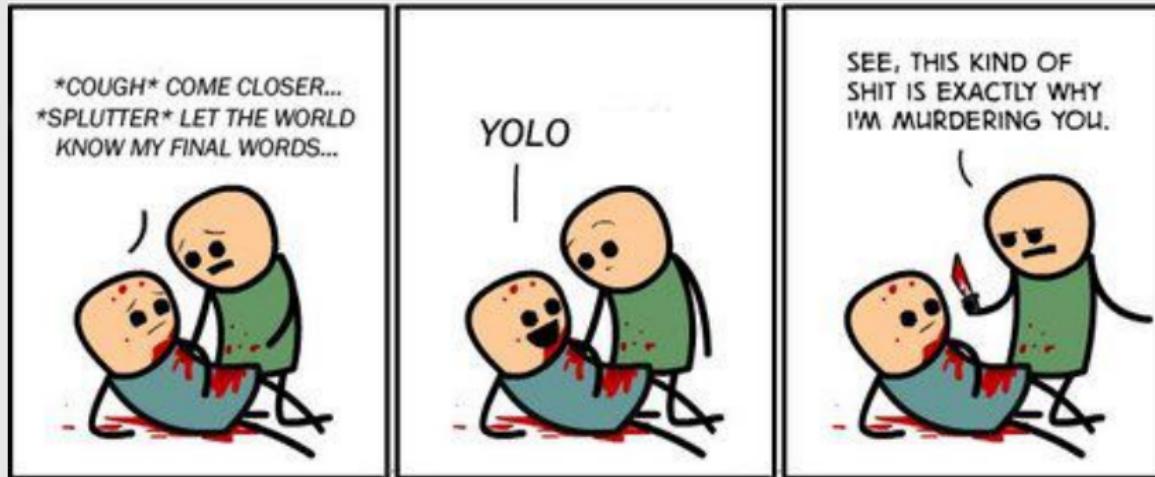
The code: Exception handling

```
procedure Set_Angle_Catch (X : Servo_Angle) is
begin
    Set_Angle (X * 2);
exception
    when Constraint_Error =>
        Put_Line ("Well, that was close");
end Set_Angle_Catch;
```

Your last chance

```
procedure Last_Chance_Handler is
begin
    -- Oops, there's something wrong
    Reset_The_Board;
end Last_Chance_Handler;
```

YOLO



¹cyanide and happiness

AdaCore

Contracts

```
procedure Set_Angle (Angle : Servo_Angle)
  with Pre => Initialized;
--  Set desired angle for the servo motor

function Initialized return Boolean;
--  Return True if the driver is initialized

procedure Initialize
  with Post => Initialized;
--  Initialize the servo motor driver
```

Null access

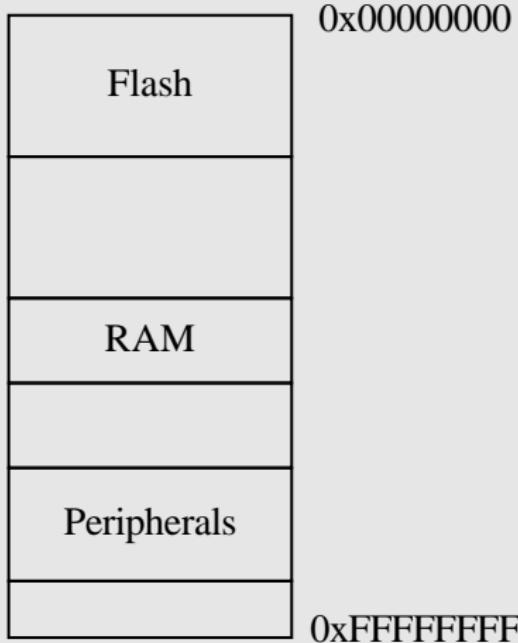
```
procedure Plop (Ptr : not null Some_Pointer);
```

Hardware mapping

```
-- High level view of the type
type Servo_Angle is range -90 .. 90

-- Hardware representation of the type
with Size      => 8,
     Alignment => 16;
```

Memory mapped registers



Hardware mapping

7	6	5	4	3	2	1	0
Reserved	Sense			Reserved			

Sense: Pin sensing mechanism

0: Disabled

2: Sense for high level

3: Sense for low level

Hardware mapping

```
#define SENSE_MASK      (0x30)
#define SENSE_POS        (4)

#define SENSE_DISABLED  (0)
#define SENSE_HIGH      (2)
#define SENSE_LOW       (3)

uint8_t *register = 0x80000100;

// Clear Sense field
*register &= ~SENSE_MASK;
// Set sense value
*register |= SENSE_DISABLED << SENSE_POS;
```

Hardware mapping

```
-- High level view of the Sense field
type Pin_Sense is
  (Disabled,
   High,
   Low)
with Size => 2;

-- Hardware representation of the Sense field
for Pin_Sense use
  (Disabled => 0,
   High      => 2,
   Low       => 3);
```

Hardware mapping

```
-- High level view of the register
type IO_Register is record
    Reserved_A : UInt4;
    SENSE      : Pin_Sense;
    Reserved_B : UInt2;
end record;

-- Hardware representation of the register
for IO_Register use record
    Reserved_A at 0 range 0 .. 3;
    SENSE      at 0 range 4 .. 5;
    Reserved_B at 0 range 6 .. 7;
end record;
```

Hardware mapping

```
Register : IO_Register  
  with Address => 16#8000_0100#;
```

```
Register.SENSE := Disabled;
```

SVD -> Ada

```
<field>
  <name>SENSE</name>
  <description>Pin sensing mechanism.</description>
  <lsb>16</lsb> <msb>17</msb>
  <enumeratedValues>
    <enumeratedValue>
      <name>Disabled</name>
      <description>Disabled.</description>
      <value>0x00</value>
    </enumeratedValue>
  [...]
```

github.com/AdaCore/svd2ada

Ravenscar Tasking

A.K.A There's a mini-RTOS in my language²

- Tasks (threads)
- Time handling
 - Clock
 - Delays
- Protected Objects:
 - Mutual exclusion
 - Synchronization between tasks
 - Interrupt handling

²blog.adacore.com/theres-a-mini-rtos-in-my-language

Task

```
task body My_Task is
    Next_Release : Time;
begin
    -- Set Initial release time
    Next_Release := Clock + Milliseconds (100);

    loop
        -- Suspend My_Task
        delay until Next_Release;

        -- Compute the next release time
        Next_Release := Next_Release + Milliseconds (100);

        -- Do something really cool at 10Hz...
    end loop;
end My_Task;
```

Making the Ada Drivers Library

Ada Drivers Library

- Firmware library
- Hardware and vendor independent
- 100% Ada
- Hosted on GitHub:

`github.com/AdaCore/Ada_Drivers_Library`

Components



↔ I2C, SPI, UART, etc.



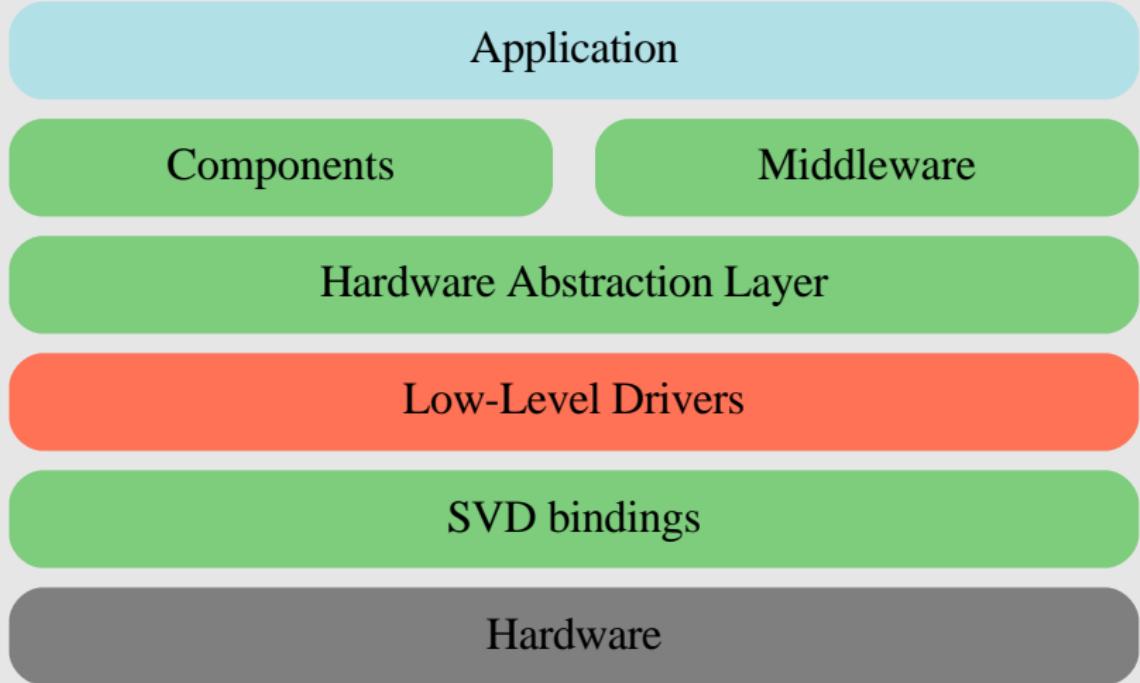
Supported components

- Audio DAC: SGTL5000, CS43L22, W8994
- Camera: OV2640, OV7725
- IO expander: MCP23XXX, STMPE1600, HT16K33
- Motion: AK8963, BNO055, L3GD20, LIS3DSH, MMA8653, MPU9250
- Range: VL53L0X
- LCD: ILI9341, OTM8009a, ST7735R, SSD1306
- Touch panel: FT5336, FT6X06, STMPE811
- Module:
 - AdaFruit's trellis
 - AdaFruit's Thermal printer

Middleware

- Bitmap drawing
- File System: FAT and ARM semi-hosting
- Log utility

Architecture



Supported platforms

ARM



STM32F405 Discovery (ARM Cortex-M4F)



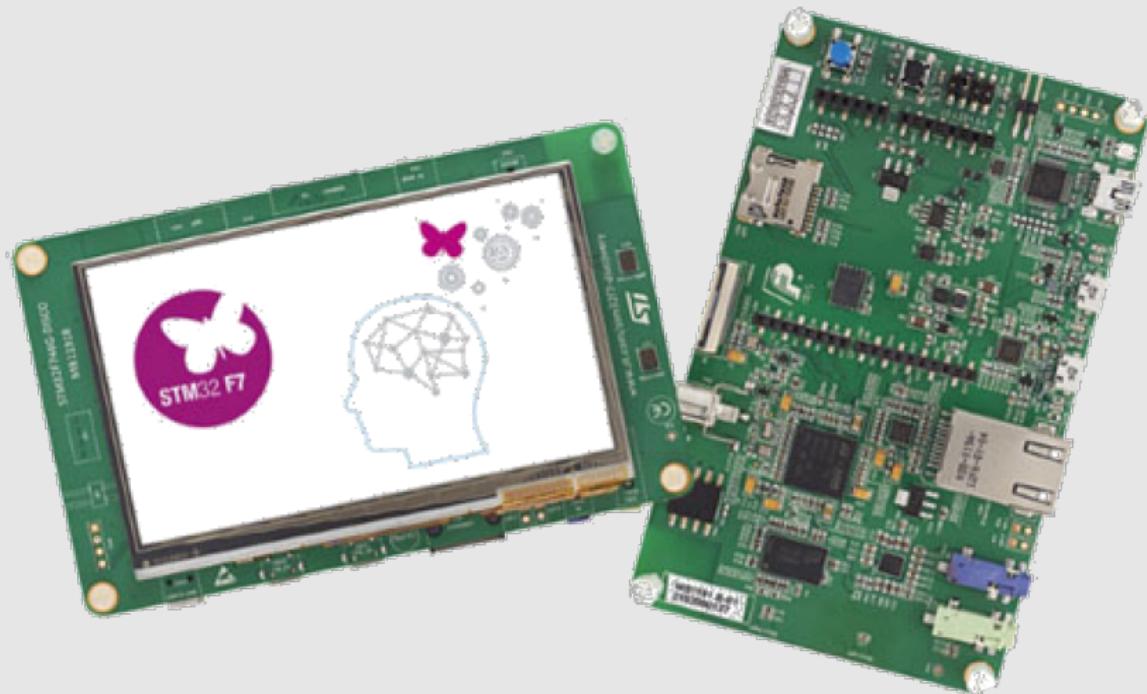
STM32F429 Discovery (ARM Cortex-M4F)



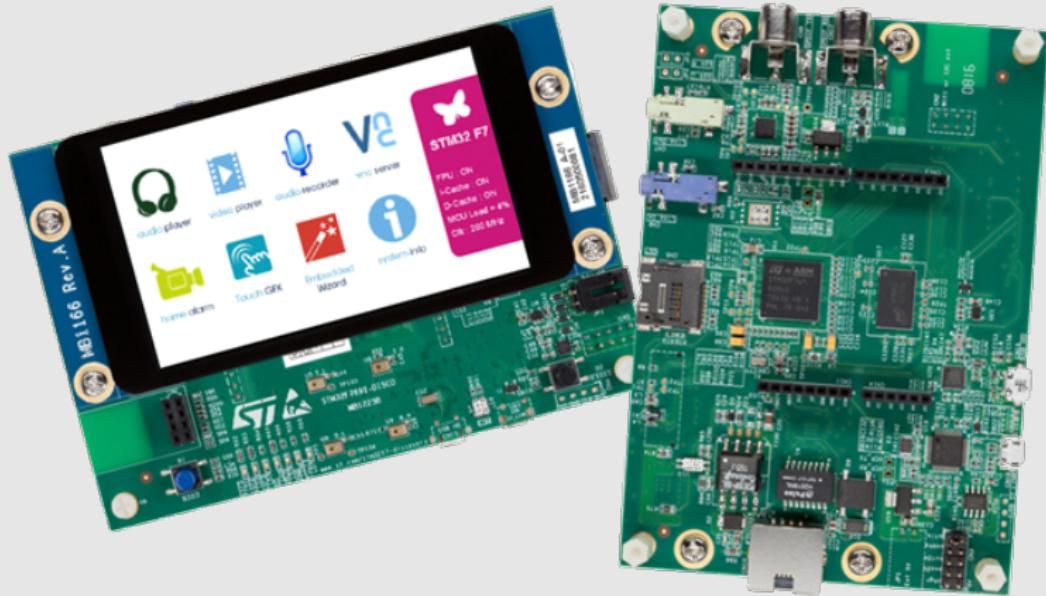
STM32F469 Discovery (ARM Cortex-M4F)



STM32F746 Discovery (ARM Cortex-M7F)



STM32F769 Discovery (ARM Cortex-M7F)



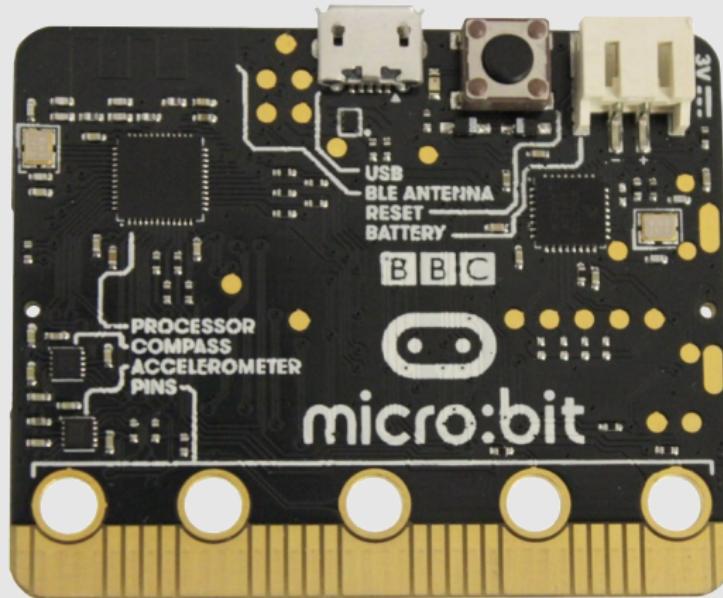
OpenMV 2 (ARM Cortex-M4F)



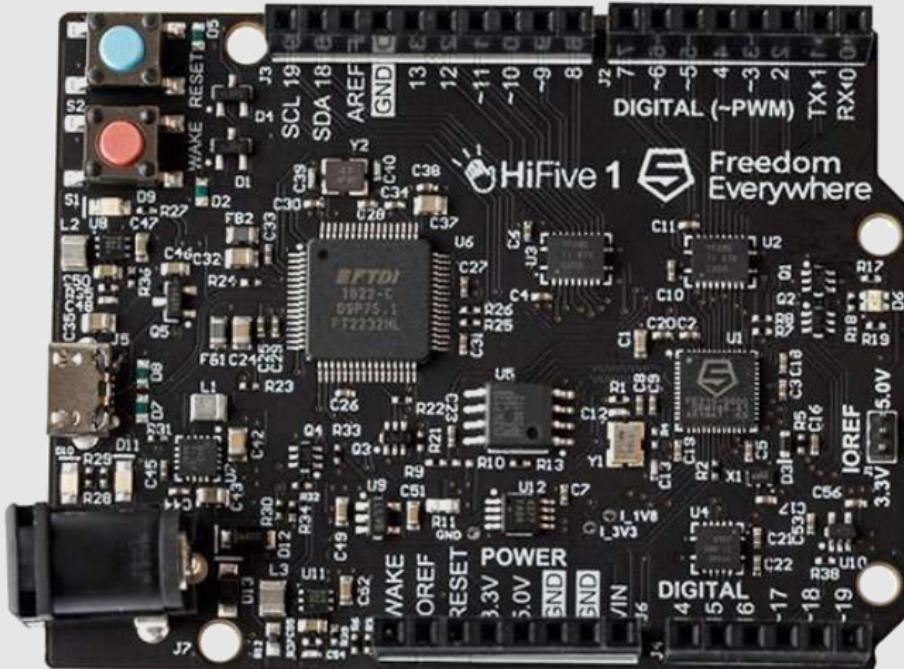
Crazyflie 2.0 (ARM Cortex-M4F)



BBC Micro:Bit (ARM Cortex-M0)



HiFive1 (RISC-V)



What's next?

TODOs:

- New configuration and build system
- More documentation
- Basic out of the box support of all the Cortex-M devices
- Linux GPIO/I2C/SPI support (on the Raspberry Pi for instance)
- AVR platform
- More component drivers
- USB stack and drivers on the STM32
- Bluetooth Low Energy stack on the Micro:Bit

Getting started demo

Download and install the tools: adacore.com/community

Download GNAT Community Edition

For free software developers, hobbyists, and students.

x86-64 GNU Linux (64 bits)

GNAT GPL Ada

[gnat-gpl-2017-x86_64-linux-bin.tar.gz](#)
SHA-1: 9682e2ef2f232ce03fe21d77b14c37a0de5649b

496.34 MB May 17 2017

SPARK Discovery

[spark-discovery-gpl-2017-x86_64-linux-bin.tar.gz](#)
SHA-1: a70d75c71508ed3ab0ecb4a34fcc1dff9a9d9089

104.06 MB May 29 2017

ARM ELF (hosted on linux)



GNAT GPL Ada

[gnat-gpl-2017-arm-elf-linux-bin.tar.gz](#)
SHA-1: 71b5830d0242dfcb294d8895960f969bbc5c2417

548.9 MB May 17 2017

Download Ada Drivers Library

This screenshot shows the GitHub repository page for AdaCore/Ada_Drivers_Library. The page includes a header with navigation links like 'Pull requests', 'Issues', 'Marketplace', and 'Explore'. Below the header, there's a search bar and a main content area for the repository. The repository details show 1,179 commits, 22 branches, 0 releases, and 15 contributors. A green progress bar indicates the repository is 100% complete. The main list of files and commits is visible, along with a 'Clone or download' section containing a 'Clone with HTTPS' button and a 'Download ZIP' button.

Ada source code and complete sample GNAT projects for selected bare-board platforms supported by GNAT.

1,179 commits 22 branches 0 releases 15 contributors BSD-3-Clause

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

File / Commit	Description	Date
.gitignore	Use new GPRbuild attribute: Create_Missing_Dirs	7 months ago
boards	robust, safer version of GPIO_PIO	5 months ago
components	SGTL5000: Fix some typos	3 months ago
docs	docs/filesystem.md: Add documentation for directory handling	3 months ago
examples	Examples: Bring back the blinky and serial examples for the STM32F4 d...	2 months ago
hal	HAL.SDMMC: Add single and multiple block write cmd definition	5 months ago
middleware	File_IO: Improve error handling	2 months ago
scripts	Examples: Bring back the blinky and serial examples for the STM32F4 d...	2 months ago
testsuite	Monitor.Block_Drivers: Show data size	5 months ago
arch	add convenience macro to conditionally enable a pin	5 months ago
hal	Merge pull request #129 from AdaCore/add_gpio_drive	5 months ago

Clone with HTTPS Use SSH
Use Git or checkout with SVN using the web URL.
https://github.com/AdaCore/Ada_Drivers_Library

Download ZIP

Some projects using the Ada Drivers Library

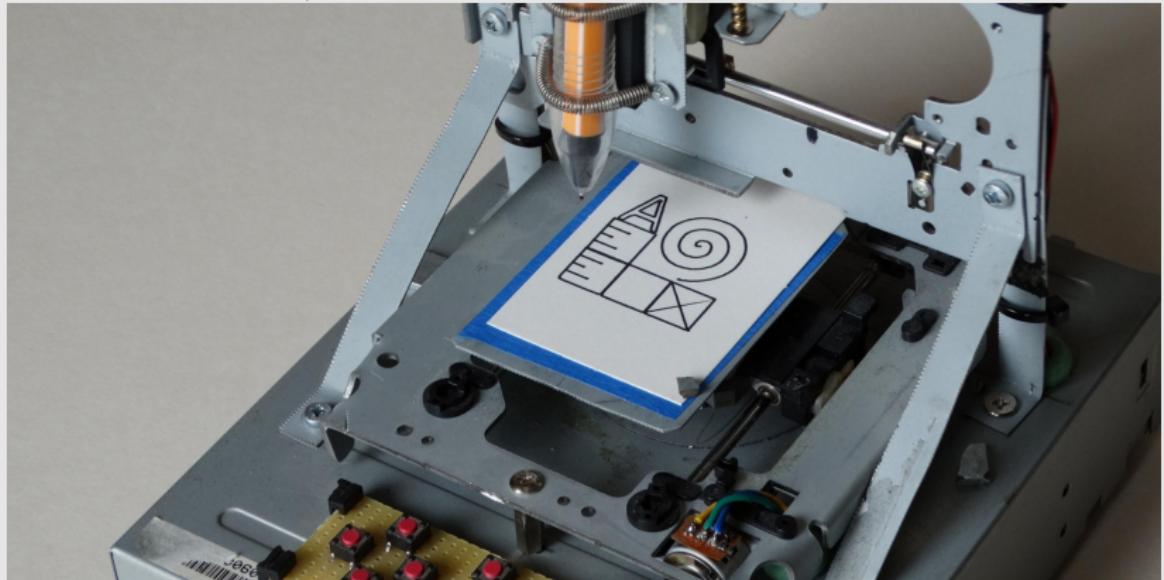
Crazyflie 2.0 Flight controller

blog.adacore.com/how-to-prevent-drone-crashes-using-spark



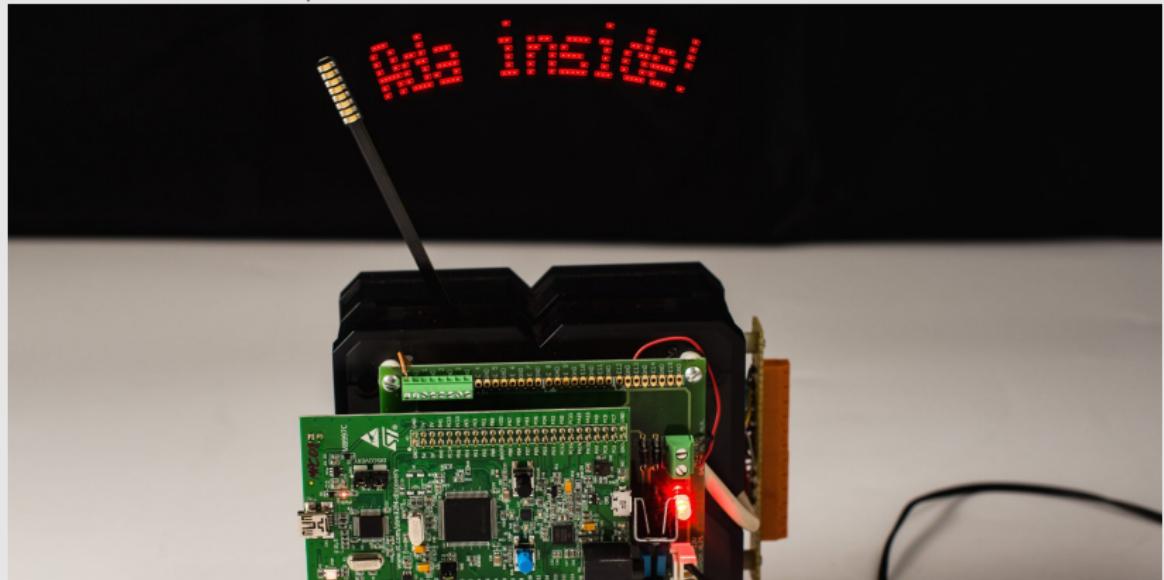
CNC Controller

blog.adacore.com/make-with-ada-arm-cortex-m-cnc-controller



Pendulum clock LED

blog.adacore.com/writing-on-air



DIY instant camera

blog.adacore.com/make-with-ada-diy-instant-camera



Wolf

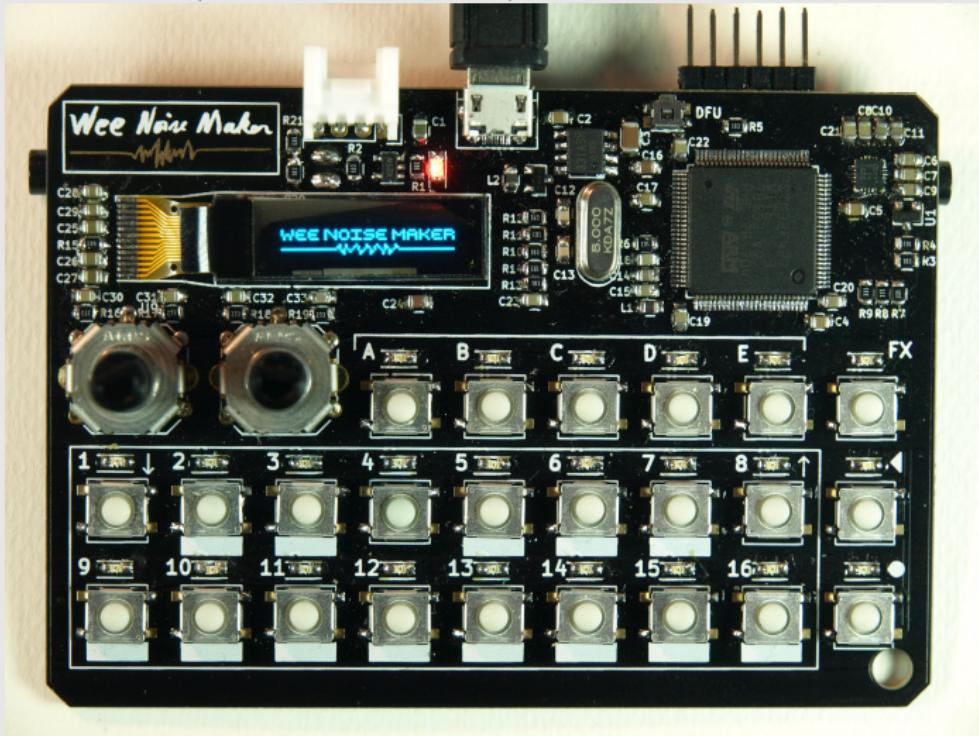
github.com/lambourg/Ada_Bare_Metal_Demos



AdaCore

Wee Noise Maker

github.com/Fabien-Chouteau/Wee-Noise-Maker



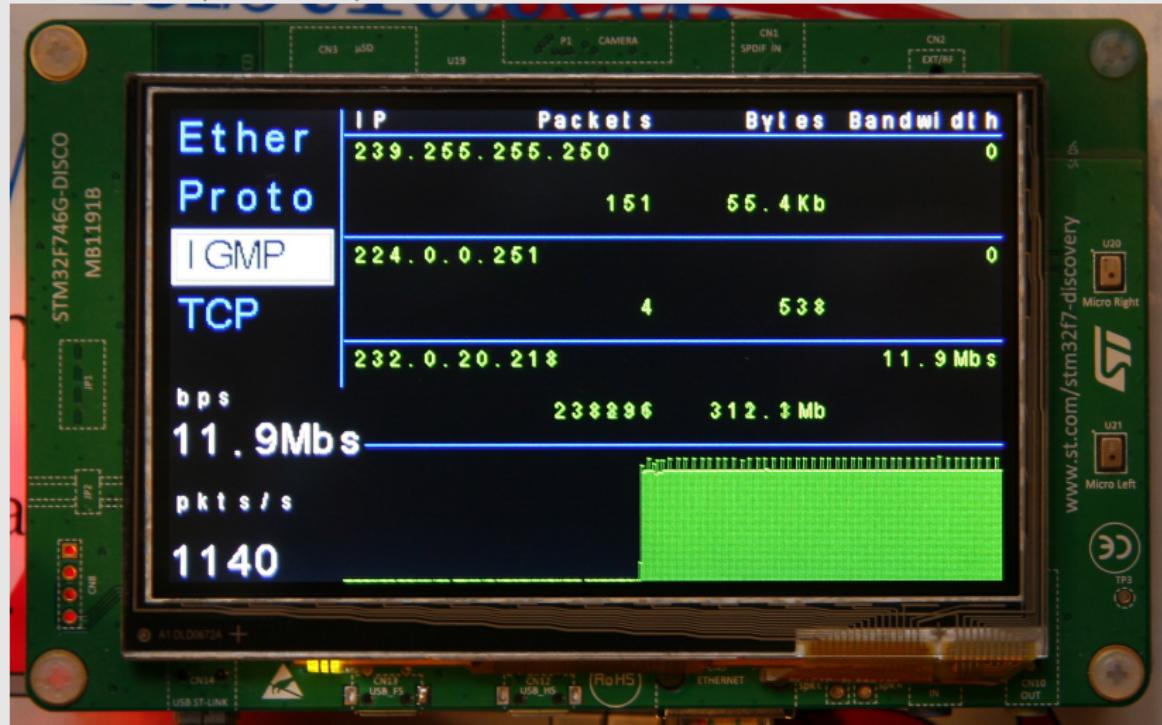
The Make with Ada Competition

- Embedded software project competition
- Open to everyone
- ~8000 euros in prize
- Stay tuned for the next edition (Twitter @adaprogrammers)

MAKE
*with***Ada**

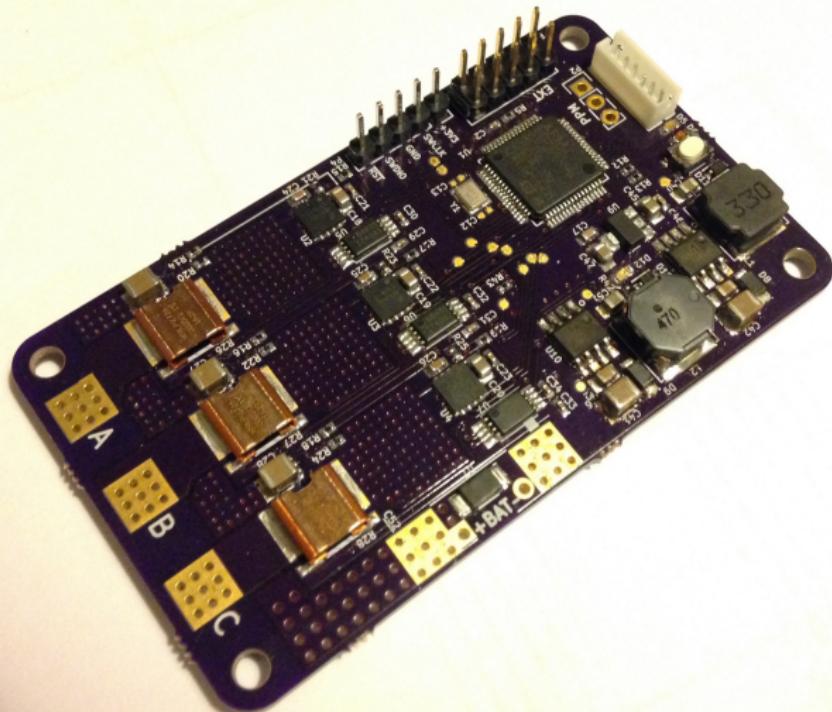
2016 Winner project (Stephane Carrez)

github.com/stcarrez/etherscope



2017 Winner project (Jonas Attertun)

blog.adacore.com/make-with-ada-2017-brushless-dc-motor-controller



What are you going to make?

- GitHub: github.com/AdaCore/Ada_Drivers_Library
- Twitter: @AdaProgrammers