Concurrent Real-Time Programming with Ada

Fabien Chouteau

Embedded Software Engineer at AdaCore

Twitter : @DesChips
GitHub : Fabien-Chouteau
Hackaday.io: Fabien.C

AdaFrance

Tasks

```
--  Task type declaration
task type My_Task (Prio : System.Priority)
  with Priority => Prio;
```

```
T1 : My_Task (Prio => 1);
T2 : My_Task (Prio => 2);
```

Time

```
task body My_Task is
   Period       : constant Time_Span := Milliseconds (100);

   Next_Release : Time := Clock + Period;
   --  Set Initial release time
begin
   loop
      --  Suspend My_Task until the Clock is greater
      --  than Next_Release.
      delay until Next_Release;
      --  Compute the next release time
      Next_Release := Next_Release + Period;

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

Mutual exclusion and shared resources

```
protected My_Protected_Object
  with Priority => 3
is

   procedure Set_Data (Data : Integer);
   --  Protected procedues can read and/or modifiy the
   --  protected data.

   function Data return Integer;
   --  Protected functions can only read the protected data

private
   --  Protected data are declared in the private part
   PO_Data : Integer := 0;
end;
```

Synchronization 1/2

```
protected My_Protected_Object is
   entry Wait_For_Signal;
   procedure Send_Signal;
private
   We_Have_A_Signal : Boolean := False;
end My_Protected_Object;
```

Synchronization 2/2

```
protected body My_Protected_Object is

   entry Wait_For_Signal when We_Have_A_Signal is
   begin
       We_Have_A_Signal := False;
   end Wait_For_Signal;

   procedure Send_Signal is
   begin
       We_Have_A_Signal := True;
   end Send_Signal;

end My_Protected_Object;
```

AdaFrance

Interrupt handling 1/2

```
protected My_Protected_Object
  with Interrupt_Priority => 255
is
   entry Get_Next_Character (C : out Character);

private
   procedure UART_Interrupt_Handler
           with Attach_Handler => UART_Interrupt;

   Received_Character  : Character := ASCII.NUL;
   We_Have_A_Character : Boolean := False;
end;
```

Interrupt handling 2/2

```
protected body My_Protected_Object is

   entry Get_Next_Character (C : out Character)
     when We_Have_A_Character
   is
   begin
       C := Received_Character;
       We_Have_A_Char := False;
   end Get_Next_Character;

   procedure UART_Interrupt_Handler is
   begin
       Received_Character  := A_Character_From_UART_Device;
       We_Have_A_Character := True;
   end UART_Interrupt_Handler;
end;
```

AdaFrance

Ada Drivers Library

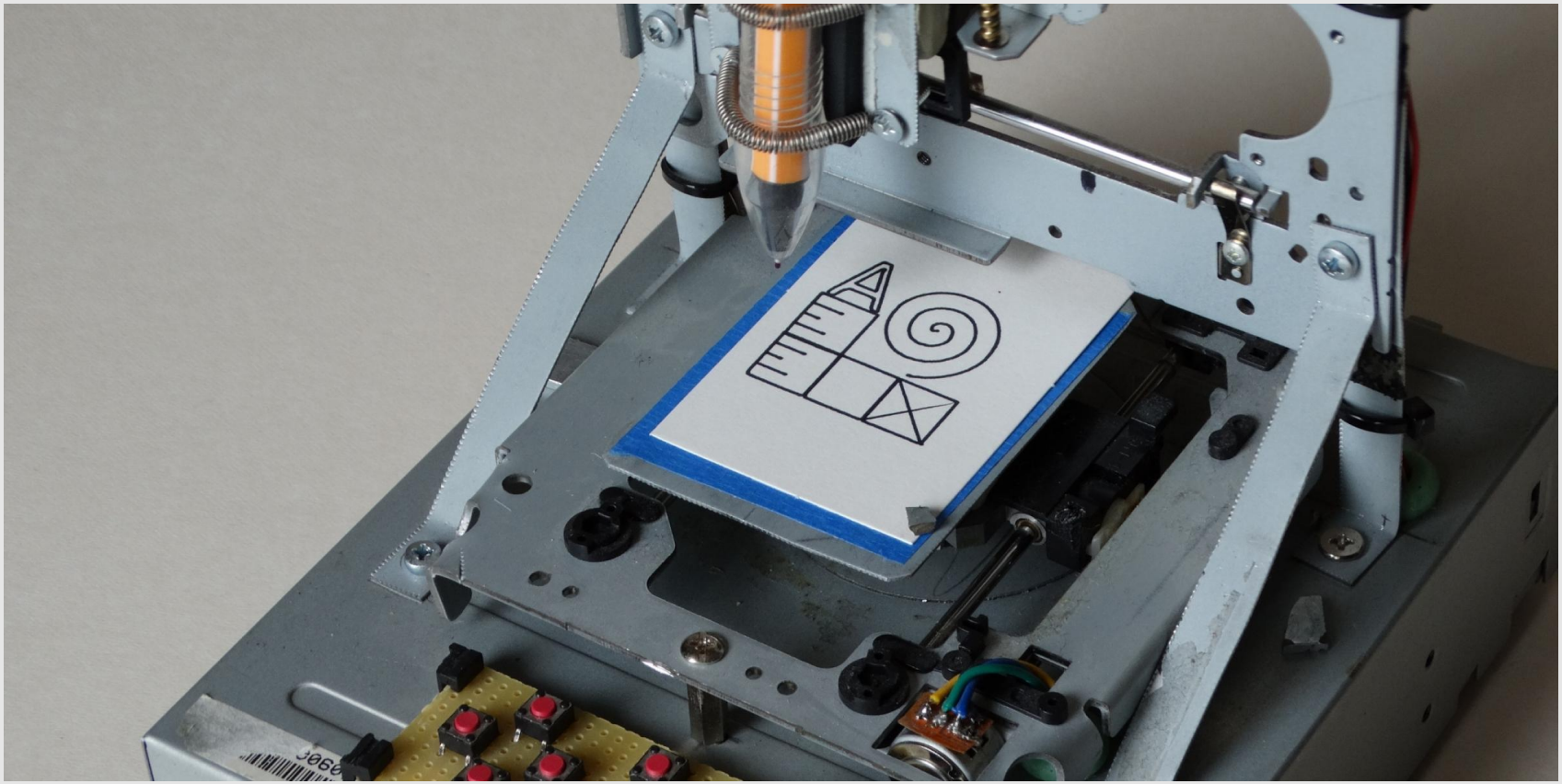
- Firmware library
- Hardware and vendor independent
- 100% Ada
- Hosted on GitHub:
  `github.com/AdaCore/Ada_Drivers_Library`

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