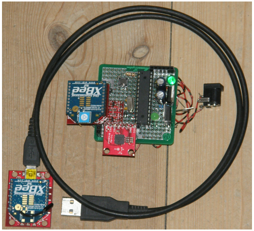
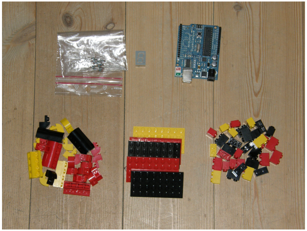
Ouverture

Programming Arduinos made even easier (and more reliable).

With AVR-Ada you can enjoy using Ada for programming embedded systems with ATMEL microcontrollers.





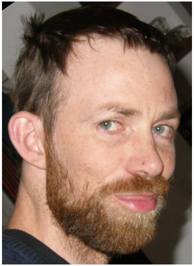
Programming Arduinos in Ada

Jacob Sparre Andersen

Jacob Sparre Andersen Research & Innovation

4th February 2012

Jacob Sparre Andersen



Currently:

- Independent consultant.
- Co-founder of AdaHeads K/S.
- Programs embedded devices for Koparo.

Background:

- PhD in experimental physics.
- BSc in mathematics.
- Has taught mathematics, physics and software engineering.
- Worked with bioinformatics, biotechnology and modelling of investments in the financial market.

jacob@jacob-sparre.dk
www.jacob-sparre.dk

Outline

1. Introduction
2. Getting started
3. Hardware
4. Software
5. A little trick for Linux users

The really short recipe

1. In parallel:
   - Get an ATMEL device (Arduino or similar).
   - Install AVR-Ada[1]
2. Compile, install and run some demonstration programs
3. Connect the device with some input or output hardware
4. Program
5. Play

[1] Ludovic, can you make it `sudo apt-get install avr-gnat` :-)

The really short recipe + links

- Make sure the device you get is a supported one:

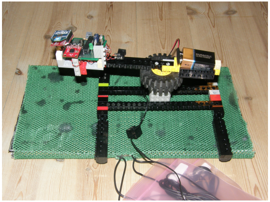
  `http://sourceforge.net/apps/mediawiki/avr-ada/index.php?title=Status`
- Official AVR-Ada installation instructions:

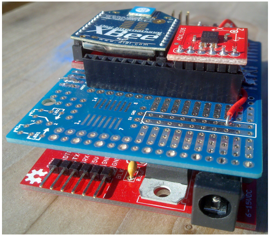
  `http://sourceforge.net/apps/mediawiki/avr-ada/index.php?title=Setup`
- Overview of distribution-specific installation guides:

  `http://arduino.cc/playground/Code/AVR-Ada`

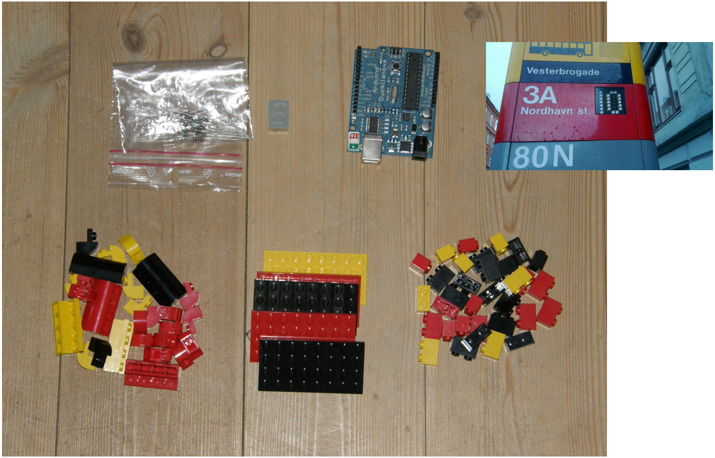
What makes the installation of AVR-Ada a bit tricky is that you need a complete matching toolset (binutils, gcc, etc.). A version for GCC 4.6 (subversion version) has just been released for the brave.

Wireless accelerometer (prototype)





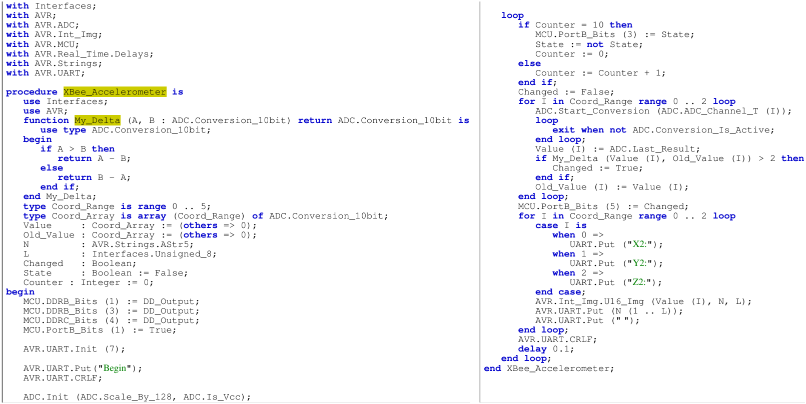
LEGO bus stop sign (in pieces)



Ada – Representation clauses

```
type LCD_Segment_Digits is (’1’, ’4’, ’7’, ’3’, ’2’,
                            ’5’, ’6’, ’0’, ’9’, ’8’);
-- MSB->LSB: top, upper left, upper right, middle,
--           lower left, lower right, bottom, decimal point
for LCD_Segment_Digits use (’0’ => 2#1_11_0_11_1_0#,
                            ’1’ => 2#0_01_0_01_0_0#,
                            ’2’ => 2#1_01_1_10_1_0#,
                            ’3’ => 2#1_01_1_01_1_0#,
                            ’4’ => 2#0_11_1_01_0_0#,
                            ’5’ => 2#1_10_1_01_1_0#,
                            ’6’ => 2#1_10_1_11_1_0#,
                            ’7’ => 2#1_01_0_01_0_0#,
                            ’8’ => 2#1_11_1_11_1_0#,
                            ’9’ => 2#1_11_1_01_1_0#);
[...]
Time_To_Next_Bus : LCD_Segment_Digits;
for Time_To_Next_Bus’Address use MCU.PortB’Address;
[...]
Time_To_Next_Bus := ’3’;
```

Ada – Wireless accelerometer



```ada
with Interfaces;
with AVR;
with AVR.ADC;
with AVR.Int_Img;
with AVR.MCU;
with AVR.Real_Time.Delays;
with AVR.Strings;
with AVR.UART;

procedure XBee_Accelerometer is
   use Interfaces;
   use AVR;
   function My_Delta (A, B : ADC.Conversion_10bit) return ADC.Conversion_10bit is
      use type ADC.Conversion_10bit;
   begin
      if A > B then
         return A - B;
      else
         return B - A;
      end if;
   end My_Delta;
   type Coord_Range is range 0 .. 5;
   type Coord_Array is array (Coord_Range) of ADC.Conversion_10bit;
   Value     : Coord_Array := (others => 0);
   Old_Value : Coord_Array := (others => 0);
   N         : AVR.Strings.AStr5;
   L         : Interfaces.Unsigned_8;
   Changed   : Boolean;
   State     : Boolean := False;
   Counter : Integer := 0;
begin
   MCU.DDRB_Bits (1) := DD_Output;
   MCU.DDRB_Bits (3) := DD_Output;
   MCU.DDRC_Bits (4) := DD_Output;
   MCU.PortB_Bits (1) := True;

   AVR.UART.Init (7);

   AVR.UART.Put ("Begin");
   AVR.UART.CRLF;

   ADC.Init (ADC.Scale_By_128, ADC.Is_Vcc);
```

```ada
   loop
      if Counter = 10 then
         MCU.PortB_Bits (3) := State;
         State := not State;
         Counter := 0;
      else
         Counter := Counter + 1;
      end if;
      Changed := False;
      for I in Coord_Range range 0 .. 2 loop
         ADC.Start_Conversion (ADC.ADC_Channel_T (I));
         loop
            exit when not ADC.Conversion_Is_Active;
         end loop;
         Value (I) := ADC.Last_Result;
         if My_Delta (Value (I), Old_Value (I)) > 2 then
            Changed := True;
         end if;
         Old_Value (I) := Value (I);
      end loop;
      MCU.PortB_Bits (5) := Changed;
      for I in Coord_Range range 0 .. 2 loop
         case I is
            when 0 =>
               UART.Put ("X2:");
            when 1 =>
               UART.Put ("Y2:");
            when 2 =>
               UART.Put ("Z2:");
         end case;
         AVR.Int_Img.U16_Img (Value (I), N, L);
         AVR.UART.Put (N (1 .. L));
         AVR.UART.Put (" ");
      end loop;
      AVR.UART.CRLF;
      delay 0.1;
   end loop;
end XBee_Accelerometer;
```

With special thanks to Tero Koskinen.

Not quite Ada

A few things are (still) missing in AVR-Ada. There is no run-time system, which means that you don't have:

- Exceptions.
- Tasking.
- Ada.Text_IO.
- Tagged types.
- . . .

The interrupt handlers are configured using a library instead of through the Ada language features.

Getting the programs unto the device

When you want to get your programs to run on your Arduino you build them, using your newly compiled AVR version of GNAT[2]:

```
avr-gnatmake -g -XMCU=atmega328p -Pblinky.gpr
```

Then you plug your Arduino to the USB port on your development host and use `avrdude` to install the program on the device:

```
sudo avrdude -c arduino -p atmega328p -P
/dev/arduino_XXXXXXXX -b 115200 -U
flash:w:blinky.hex
```

[2] Make sure you identify the right processor type. Arduino UNO r3 uses atmega328p.

A little trick

If you want your Arduino to be named consistently, placing a rule like this in a file in `/etc/udev/rules.d/` will be useful:

```
SUBSYSTEMS=="usb", ATTRSidVendor=="0403",
ATTRSidProduct=="6001", ATTRSproduct=="FT232R
USB UART", KERNEL=="ttyUSB*",
ATTRSserial=="A700eEq1",
SYMLINK+="accelerometer_1", MODE:="0660",
GROUP:="koparo"
```

(all on one line)

Contact

Jacob Sparre Andersen
+45 21 49 08 04

Jacob Sparre Andersen
Jacob Sparre Andersen Research & Innovation
`jacob@jacob-sparre.dk`
`http://www.jacob-sparre.dk/`

AVR-Ada on Sourceforge
`http://avr-ada.sourceforge.net/`

The Arduino web site
`http://arduino.cc/`

Tero Koskinen
`tkoskine` in the `#Ada` IRC channel on Freenode