MuntsOS Embedded Linux

Application Note #7: Ada LED Flash Example Using Alire

Revision 7
13 December 2024

by Philip Munts
dba Munts Technologies
http://tech.munts.com

Introduction

This application note describes how to create, build, and run an Ada program to flash an LED on a target computer running ***MuntsOS Embedded Linux***, using the ***Alire*** (Ada LIbrary REpository) development environment (https://alire.ada.dev).

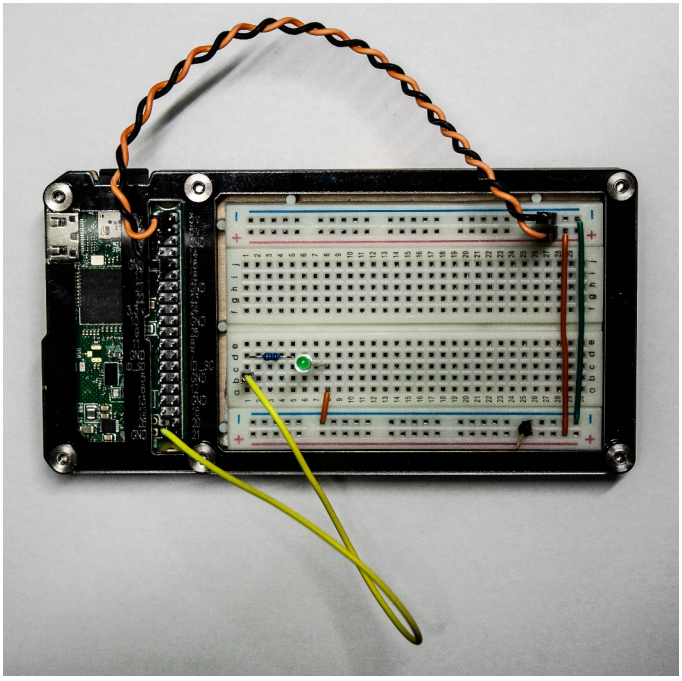
Prerequisites

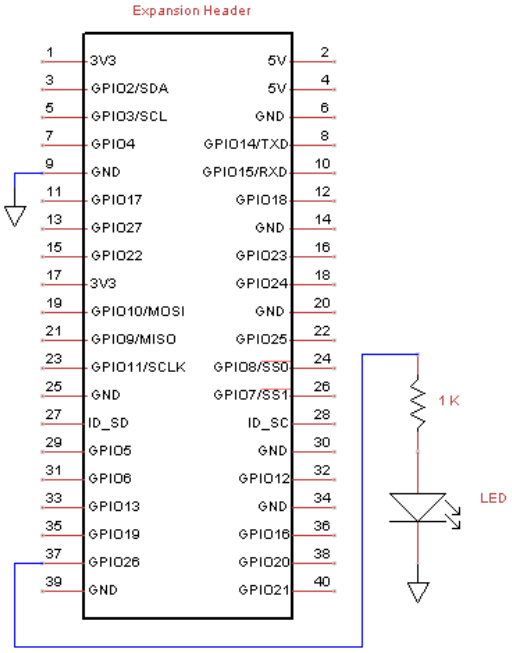
The ***MuntsOS Embedded Linux*** software development environment must be installed on a Linux development computer (AppNote #1 or AppNote #2).

The **`alr`** command line tool for ***Alire*** must be installed on the desktop Linux system (following the procedure specified at https://alire.ada.dev/docs/#installation.

MuntsOS Embedded Linux must be installed on the target computer (AppNote #3).

Target Platform Hardware





The test platform for the purposes of this application note consists of a Raspberry Pi Zero 2 Wireless mounted in a Zebra Zero Plus Breadboard case. The orange and black jumper wires connect **`+3.3V`** and **`GND`** on the Raspberry Pi expansion header to the breadboard power rails. The yellow jumber connects **`GPIO26`** to a 1K ohm current limiting resistor and an LED.

Test Program Source Code

Available for download at: http://git.munts.com/muntsos/doc/.blinky/blinky.adb

```
WITH Ada.Text_IO; USE Ada.Text_IO;

WITH GPIO.libsimpleio;
WITH RaspberryPi;

PROCEDURE blinky IS

  LED : GPIO.Pin;

BEGIN
  New_Line;
  Put_Line("MuntsOS Ada LED Test");
  New_Line;

  -- Configure a GPIO output to drive an LED

  LED := GPIO.libsimpleio.Create(RaspberryPi.GPIO26, GPIO.Output);

  -- Flash the LED forever (until killed)

  Put_Line("Press CONTROL-C to exit");
  New_Line;

  LOOP
    LED.Put(NOT LED.Get);
    DELAY 0.5;
  END LOOP;
END blinky;
```

Exercise

This example exercise demonstrates how to create an Alire Ada program project for ***MuntsOS Embedded Linux***, compile it, and run it on the test platform hardware.

Step 1: Prepare the **`blinky`** project directory:

```
alr -n init --bin blinky
cd blinky
alr -n with muntsos_aarch64
wget -O src/blinky.adb https://tinyurl.com/blinkprogs/blinky.adb
ALIRE_DISABLESTYLECHECKS=yes alr action -r post-fetch
```

The above commands will:

- Create and initialize an Alire project directory.
- Add a reference to the **`muntsos_aarch64`** library crate to the project.
- Download the source program **`blinky.adb`**.
- Run the **`muntsos_aarch64`** library crate postfetch script.

The postfetch script performs the following actions:

- Converts the project into a cross-compiled application for ***MuntsOS Embedded Linux*** running on a 64-bit AArch64 target computer, which the Raspberry Pi Zero 2 Wireless is.
- Disables GNAT style checks in **`blinky.gpr`** if the **`ALIRE_DISABLESTYLECHECKS`** environment variable is set to **`yes`**. This is optional, but without it, the compiler may emit style warnings when you build the project.

Step 2: Build the **`blinky`** project:

```
alr build
```

Step 3: Copy **`blinky`** to the test platform:

```
scp bin/blinky root@snoopy:.
```

Step 4: Run the test program on the test platform:

```
ssh root@snoopy
./blinky
```

The LED should begin flashing once a second, until you press **`CONTROL-C`**.