Project Arduino A2
Technical Documentation

The Project Arduino A2 Team

October 21, 2025

Contents

1 Overview

Project Arduino A2 is an open-source embedded application for the Arduino Uno platform. It demonstrates a sensor-rich environment that reads ambient humidity/temperature (DHT11), light intensity (BH1750), and GPS telemetry (GY-GPS6MV2), while driving a multi-colour RGB LED with a non-blocking animation. The code base is organised into reusable modules for each device, enabling the addition or removal of sensors with minimal changes to the firmware entry point.

2 Repository Structure

Path	**Description**
`platformio.ini`	PlatformIO configuration targeting Arduino Uno.
`src/main.cpp`	Firmware entry point that delegates to module initialisation and update functions.
`src/actuators/rgb/`	RGB LED actuator module interface and implementation.
`src/sensors/dht/`	DHT11 temperature and humidity sensor module.
`src/sensors/bh1750/`	BH1750 ambient light sensor module.
`src/sensors/gps/`	GPS (GY-GPS6MV2) module powered by TinyGPS++.
`docs/project_documentation.tex`	This documentation file.

3 Hardware Requirements

3.1 Bill of Materials

- Arduino Uno (or compatible ATmega328P board).
- DHT11 temperature and humidity sensor (with 10 k Ω pull-up if using the bare element).
- BH1750 digital light sensor (I^{2}C breakout).
- GY-GPS6MV2 GPS module (u-blox NEO-6M based).
- Common-cathode RGB LED (connected via suitable current-limiting resistors).
- Assorted jumper wires and breadboard or PCB.

3.2 Pin Assignments

Device	**Arduino Pin**	**Notes**
`DHT11 DATA`	D2	configured with internal pull-up in firmware.
`BH1750 SDA`	A4	4.7 k–10 kΩ pull-up to VCC typically provided on breakout.
`BH1750 SCL`	A5	shares the pull-up with SDA.
`GPS TX`	D4	feeds SoftwareSerial RX; keep wiring short.
`GPS RX`	D3	optional; level-shift to 3.3 V if used.
`RGB LED Red`	D9	PWM capable.
`RGB LED Green`	D10	PWM capable.
`RGB LED Blue`	D11	PWM capable.
`VCC / GND`	5 V / GND	shared across all modules.

4 Software Setup

4.1 Tooling

1. Install PlatformIO IDE or the PlatformIO Core CLI.
2. Ensure a recent Arduino toolchain is available via PlatformIO (handled automatically).
3. Optionally install a LaTeX distribution (TeX Live, MikTeX) to compile this document.

4.2 Dependencies

Dependencies are declared in `platformio.ini` under `lib_deps`:

- `adafruit/DHT sensor library@`
  `1.4.6`
- `adafruit/Adafruit Unified Sensor@`
  `1.1.15`
- `claws/BH1750@`
  `1.3.0`
- `mikalhart/TinyGPSPlus@`
  `1.1.0`

PlatformIO automatically downloads these libraries during the first build.

4.3 Building and Uploading

1. Connect the Arduino Uno via USB.
2. From the project root run `pio run` to compile.
3. Upload using `pio run -t upload`; optionally specify `-upload-port` if auto-detection fails.
4. Launch the serial monitor with `pio device monitor -baud 9600` to observe sensor output.

5 Module Overview

5.1 RGB LED Actuator

- Location: `src/actuators/rgb/`.
- Provides `rgbInit()` and `rgbUpdate(now)`.
- Implements a colour cycle with configurable on/off timing using PWM outputs 9–11.
- Designed to be non-blocking; timing uses the `millis()` value supplied by the caller.

5.2 DHT11 Sensor

- Location: `src/sensors/dht/`.
- Provides `dhtInit()` and `dhtUpdate(now)`.
- Reads humidity and temperature every two seconds, printing results or reporting read failures.
- Uses the internal pull-up and initialises with a warm-up delay to improve stability.

5.3 BH1750 Sensor

- Location: `src/sensors/bh1750/`.
- Provides `bh1750Init()` and `bh1750Update(now)`.
- Sets the BH1750 into continuous high-resolution mode and reports lux readings once per second.
- Handles communication errors gracefully by logging failures to `Serial`.

5.4 GPS Module

- Location: `src/sensors/gps/`.
- Provides `gpsInit()` and `gpsUpdate(now)`.
- Uses SoftwareSerial on pins 4/3 to read NMEA sentences and TinyGPS++ to parse them.
- Prints satellite count, HDOP, UTC time, and fix status every second.
- Recommends taking the module outdoors for the first fix.

6 Extending the Project

6.1 Adding a New Sensor or Actuator

1. Create a new sub-directory under `src/sensors/` or `src/actuators/`.
2. Define a header exposing `init` and `update` functions.
3. Implement non-blocking behaviour so the main loop stays responsive.
4. Include the module header in `src/main.cpp` and invoke the new functions inside `setup()` and `loop()`.
5. Add any required libraries to `platformio.ini`.

6.2 Testing Suggestions

- Use PlatformIO unit testing (`pio test`) for logic that can be abstracted away from hardware.
- For hardware regression, consider logging data to an SD card or host script for repeatable validation.
- When multiple modules print to Serial, prefix each line for easier parsing or use a structured logging format.

7 Contribution Guidelines

- Follow the repository directory conventions when adding new modules.
- Prefer non-blocking code that cooperates with the existing timing strategy.
- Submit pull requests with a summary of changes, hardware tested, and any known limitations.
- Keep comments concise; focus on explaining non-obvious logic or hardware assumptions.

8 Licensing

Choose an open-source license that matches your goals (e.g., MIT, Apache-2.0, GPL-3.0). Place the licence text in a `LICENSE` file at the repository root and reference it here once selected.

9 Document Compilation

Compile this document with:

```
pdflatex docs/project_documentation.tex
```

Run the command twice to ensure references and the table of contents are up to date.

10 Acknowledgements

This project builds on community-maintained libraries from Adafruit, the TinyGPS++ authors, and contributors to the BH1750 driver. We appreciate their work and encourage contributions back to upstream projects when improvements are made.