ELEKTRONICA-ICT

Bachelorproef 2024-2025

Real-Time Multi-Sensor Logger

Author *Mauro De Bruyn*

Abstract

This application note presents a wireless, real-time multi-sensor data logger using the PSoC 6 Pioneer Kit (CY8CKIT-062-WiFi-BT). The system acquires digital, analog, and I²C signals, timestamps them with the on-chip RTC, and sends measurements to Firebase Realtime Database in JSON format, while maintaining a local flash backup during connectivity loss. It leverages the board's internal hardware the ambient light sensor (ALS), motion sensor (IMU over I²C), and a PDM microphone with PCM conversion and uses the integrated Wi-Fi module to perform periodic (every 10 s) HTTP/HTTPS uploads. A TFT display shows live sensor values. A green LED indicates a healthy cloud connection and a red LED indicates a network error, in which case records are buffered to local flash until the connection is restored.

Content

Met opmerkingen [Bart Stuk1]: Alle genummerde titels komen hier. Beperk in overdosis van titels en maximale aangeraden diepte is 3 niveaus, maar meeste kunnen zich best beperken tot 2 niveaus.

1 Introduction

This work addresses three recurring constraints in connected sensing: Multiple sensor values must be sampled and time aligned, loss of internet must not lead to data loss and data needs to be available in real time and be saved to a cloud.

This application note addresses these constraints by creating a real-time wireless muti sensor data logger build on the PSoC 6 Pioneer Kit (CY8CKIT-062-WiFi-BT) that integrates on board sensors with cloud logging to Firebase Realtime Database.

PSoC offers integrated ADC's, I²C, PDM to PCM audio conversion, an RTC, and Wi-Fi connectivity so its ideal for this project. Firebase provides a low latency real-time database that is free to use and ideal for prototyping. However, what is most important is robust operation under network faults and efficient use of memory.

Met opmerkingen [Bart Stuk2]: Minimaal 150 woorden en aangeraden 300 woorden (meer mag).
Te schrijven tegen fase 1 en 4 (Bachelorproef).

1.1 Research question and objectives

The research question is: *How can a PSoC be designed to acquire ambient light, motion, and sound along with I²C and digital/analog sensors. Timestamp the measurements and stream them as JSON to a cloud database in real time, while guaranteeing no data loss during network outages?*

Objectives:

1. Implement an acquisition of the on-board ambient light sensor (ALS via ADC), IMU (via I²C), and a PDM microphone (PCM).
2. Provide constant timestamping using the integrated RTC.
3. Upload measurements at a fixes interval (every 10 sec) to Firebase in JSON format.
4. Ensure redundancy by buffering the data using the on-chip flash memory during connectivity failures.
5. Provide a local user interface via the TFT display and status LED.

1.2 Scope and limitation

The design prioritizes the internal peripherals of the CY8CKIT-062-WiFi-BT. Wireless transport targets Wi-Fi with HTTP/HTTPS transactions to Firebase, Bluetooth, MQTT brokers or alternative cloud storage falls out of scope. Power profiling, advanced security and cloud based dashboards are noted as possible future work. The Local UI is minimal to keep focus on the logging.

1.3 Starting point

The implementation is based on the vendors documentation for the PSoC 6 and ModusToolbox, the CY8CKIT-062-WiFi-BT board, and Firebase Realtime Database developer. These sources provide information about the peripheral configuration (ADC, I²C, PDM/PCM), RTC use, and Wi-Fi connectivity.

2 Material and methods

Met opmerkingen [Bart Stuk3]: Minimaal 750 woorden en aangeraden 2000 woorden (meer mag).
Te schrijven tegen fase 1 (Bachelorproef).

2.1 Materials

2.1.1 Hardware

- Platform: Infineon PSoC 6 Wi-Fi BT Pioneer Kit (CY8CKIT-062-WiFi-BT) with on board Murata Wi-Fi/BT module, QSPI NOR flash, KitProg debugger, and Arduino compatible headers. This kit is used as the “brains” or so called connectivity hub.
- Display & sensor shield: CY8CKIT-028-TFT “TFT Display Shield Board” providing a 2.4” TFT display, motion sensor (IMU), ambient light sensor (ALS), and PDM microphone, plus an audio codec. The display uses a parallel (8080-style) interface while sensors interface via I²C, analog, or PDM.
- Optional external sensors: Analog or digital GPIO sensors and I²C peripherals can be added through the Arduino headers when needed for specific measurements.

2.1.2 Software

- IDE/SDK: ModusToolbox for project creation, BSP’s, and middleware (HAL, Wi-Fi, FreeRTOS, PDL)

- Cloud backend: Firebase Realtime Database accessed over HTTP, JSON used for structured messages.
- Firmware: PSoC 6 HAL drivers (ADC, I²C, PDM/PCM, GPIO, RTC), Wi-Fi middleware, and JSON.

2.2 Methods

2.2.1 Sensing and sampling

- Ambient Light (ALS → ADC): Raw ADC counts are converted to a LUX value via a linear calibration.
- Motion (IMU → I²C): The on-board IMU (BMI160) is initialized over I²C, raw accelerometer and gyroscope values are measured and converted to the standard units.
- Sound (PDM mic → PCM/PDM): The PDM/PCM block performs hardware decimation to 16-bit PCM. The firmware computes a short-term RMS and reports a “sound” metric (RMS).

2.2.2 Timekeeping

- RTC: The on-chip RTC is initialized at boot. The user can set the correct date and time of the RTC block in the device configurator window.

2.2.3 Data

Each upload transmits a compact JSON document with the following layout.


```
{
  "timestamp": "2025-07-20T14:30:00Z",
  "als_lux": 320.1,
  "imu": { "ax": 0.03, "ay": 0.01, "az": 9.79, "gx": 0.1, "gy": 0.2, "gz": -0.1 },
  "sound_rms": 65.2,
}
```


All of the records are saved in the *sensor/data/* path, all of these measurements are saved in a different document in this folder every ten seconds. The document does not overwrite, this was made to be able to read measurement history.

2.2.4 Cloud and redundancy

- Upload period: Uploads are performed periodically, every ten seconds. The latest sensor values are stored in a JSON file and send to the Firebase Realtime Database using a HTTP/HTTPS POST.
- Error handling: On a network or HTTP error, the record is saved to the local flash memory of the PSoC 6.
- Security: WPA2-PSK for Wi-Fi, HTTP certificates is preferred but not necessary when using firebase in developer mode (not recommended).

2.2.5 Local user interface

- TFT: minimal live readouts (ALS, motion sensor, sound RMS). The parallel TFT interface avoids I²C connection problems with the sensors.
- Status LED: Green indicates an active connection while red indicates a network error. This means that the data will be kept local until the connection is restored.
- Diagnostic: The serial monitor (UART) exposes logs, debug data, and basic commands that are happening.

2.2.6 Firmware

1. Sensor task: Acquire ALS/IMU/Audio, and store these values.

2. RTC: Get RTC value data and store it.
3. Combine: Every ten seconds, combine all of the data including the RTC value to a JSON format.
4. Upload: Try REST upload, on failure, store in flash memory, on success, advance seq and drain pending entries.
5. UI Task: Update the TFT display with accurate values and check for network status (for UI LED).

A state machine checks the connectivity: *INIT → WIFI_CONNECT → RUN (online/offline).* Transitions set the LED state and disable/enable the flash memory.

2.3 What was not used and why

- Alternative clouds and MQTT were not selected as the focus of this project lies on the PSoC and not databases. Firebase has a shallow learning curve and is therefore a great solution for prototyping work
- Audio playback was not in the scope of work, the only output for the shield is the TFT screen.
- Bluetooth was not used due to a continuous nature of logging and need for straightforward IP connectivity

3 Results

Met opmerkingen [Bart Stuk4]: Minimaal 250 woorden en aangeraden 1000 woorden (meer mag).
Te schrijven tegen fase 2 en 3 (Bachelorproef).

This section is about the objective outcomes of the implementation on the PSoC 6 Pioneer Kit combined with the CY8CKIT-028-TFT shield. The sensors were split and programmed individually, later on the sensors were implemented one by one.

3.1 Sensor capture

3.1.1 Ambient Light (ALS → ADC)

The analog ALS channel was sampled and converted to lux. Across the overall readings the ADC output showed stable values without clipping.

3.1.2 Motion sensing (IMU → I²C)

The IMU on the shield was initialized over I²C. It continuously samples data from the accelerometer and gyroscope streams. During operation, the I²C bus created some problems, the shield uses multiple pairs of I²C pins and not all of them are looped trough. This mis-routing resulted in zero readings. After selecting the correct I²C pins (the shield exposes multiple I²C header pairs), measurements were restored.

3.2 Time stamping

The on-chip RTC provided continuous timekeeping. Manual RTC setting introduced a noticeable delay before the updated value appeared on the UART console. On the final version the real-time clock does not have these problems and syncs every hour to ensure accurate time data.

3.3 Periodic cloud upload

The system produces JSON to be sent to the database every ten seconds. This contains ALS, IMU and audio metrics, alongside metadata like RTC timestamps. The database was easy to set up but the connection with the PSoC did not go smoothly, the HTTPS client failed to build/run in the MTB environment.

3.4 Local User interface

The TFT screen shows numeric readouts for light, motion and magnitude, and sound RMS. The LED indicator was integrated as the connection with the database failed.

3.5 Local fallback memory

When the network connectivity gets lost the JSON data will be KV-stored to the QSPI flash memory. Every JSON file is kept until the network connects again. Then the data will all be send to the Firebase Realtime Database and the flash memory will be dumped to use again if needed.

4 Conclusion

Met opmerkingen [Bart Stuk5]: Minimaal 150 woorden en maximaal 300 woorden.
Te schrijven tegen fase 4 (Bachelorproef).

This application note set out to realize a wireless, real-time data logger on the PSoC® 6 Pioneer Kit with the CY8CKIT-028-TFT shield. Getting ALS/IMU/PDM signals, timestamping via RTC and uploading JSON records to Firebase with a local flash backup fallback.

I did come across major problems as even vendor template projects were difficult or unable to run, and combining working code proved instable. Typical failures included errors in libraries and wrong pin configurations. These factors slowed the development and unfortunately made the project incomplete.

To reduce these problems in the future I recommend:

- Importing only the required libraries and components as some libraries make components fail.
- Integrate incrementally: start from a basic project and add ADC → IMU → PDM/PCM → Wi-Fi → Firebase in that order

In summary, the objective is achievable on the chosen platform, but the results depend on the software assembly more than sensor or network APIs.