ECE H371 Project Activities and Progress

Team Members: Ayush Nadiger, John Dariotis, Shawn Joseph, Luke Rattanavijai

---

Introduction to Cellular IoT

Cellular IoT refers to the use of cellular networks to connect and manage Internet of Things (IoT) devices. Unlike IoT devices that rely on short-range communication technologies like Wi-Fi or Bluetooth, cellular IoT devices leverage existing cellular networks (like 4G LTE and 5G) to transmit data over long distances. This enables widespread connectivity, mobility, and scalability for IoT applications.

Common Technologies under Cellular IoT:

- **NB-IoT (Narrowband IoT):** Designed for low-power devices with minimal data requirements, ideal for applications like smart meters and environmental sensors.
- **LTE-M (LTE Cat-M1):** Supports moderate data rates and mobility, suitable for wearables and asset tracking.
- **5G IoT:** The latest generation, offering enhanced features like ultra-low latency and massive device connectivity.

Importance of Time Synchronization

Time synchronization is a fundamental requirement for the effective operation of cellular IoT devices. It ensures that data collected from diverse sources can be accurately combined and analyzed, operations are performed in a coordinated manner, and network resources are utilized efficiently. Without proper time synchronization, IoT systems can suffer from:

- Data inconsistencies
- Operational inefficiencies
- Security vulnerabilities
- Increased energy consumption

To address these challenges, our project focused on synchronizing time on 5G cellular IoT boards with an NTP (Network Time Protocol) server, evaluating the synchronization algorithm under various conditions. We go over this in further detail in our 1 page research paper.

10/17/2024 : Connect given board with Internet and confirm your connection

Initial Setup and Compilation

- **Downloaded Expansion Pack:** We began by downloading the necessary expansion pack for the project.
- **Conversion to C/C++ Project:** The project was converted to a C/C++ style to allow for successful builds.
- **Build Attempts:** Initial build attempts encountered errors, suggesting issues with the setup.
- **Missing Linker Files:** Upon meeting with a PhD student mentor, we identified missing linker files in the extension pack.
- **Reinstalled Expansion Pack:** After reinstalling the expansion pack, the build was successful.

Exploration and Configuration

- **File Review:** Following guidance from the PhD mentor, we reviewed the file app_netxduo.c, analyzing its functions and their purposes. We also explored multiple APIs provided for the board.
- **Running the Project:** With the configured files, we ran the project using TeraTerm (switched from Putty due to display issues) and successfully retrieved the board's IP address.
- **Connecting to the Internet:** The board was connected to the Internet.

10/31/2024 : Establish connection with NTP server

NTP Server Connection and Debugging

- **Connection to NTP Server:** Edited the app_netxduo.c file to implement functions for connecting to the NTP server.
- **Function Review:** Focused on the sntp_time_sync_internal() function.
- **Drift Calculation:** Modified the app_main_thread_entry function to calculate time drift. Initial attempts indicated errors in the drift calculation.
- **System Time Retrieval:** Tried using GetTimeInSeconds() (undefined in the file) and switched to alternative methods like time and TIM2->CCR2 + TIM2->CNT. A problem we encountered was that the header file “tim.h” was not included in the project despite being in the file system, and we had to figure out how to include it to use this timer module.
- **Successful Connection:** The board was successfully connected to the 0.pool.ntp.org and time.google.com SNTP servers and attempted to synchronize time with the system clock. Drift calculation was returning unsigned long, changed to signed long instead.

Current Status

- Successfully connected the board to an SNTP server.
- Observed that SNTP time increased very slowly compared to system time, which used the board's timer and updated multiple times before the next SNTP time update.
- Drift expression: SNTP_time - system_time
- Realized the need to match a single system_time with a corresponding SNTP_time rather than multiple system_times with one SNTP_time.

Future Improvements: If more time were available, we would analyze the patterns between the different time values to develop a more accurate and cohesive mathematical expression for the drift between the SNTP server and the system time.

Acknowledgements

We would like to express our sincere gratitude to:

- **Professor Raza :** For providing valuable guidance and support throughout the project.
- **PhD Student Mentor, Murari :** For offering insights into debugging, reviewing our implementation, and helping us troubleshoot critical issues.

Their expertise and encouragement were invaluable in helping us try to complete this project.