NOTES

This project addresses two key areas:

- Printf is not thread-safe when used with FreeRTOS. While output remains intact if print requests originate from a single task, concurrent print operations from multiple tasks may result in garbled or overwritten messages. Several approaches exist to mitigate this issue; I have selected the more complex method to ensure robust protection.
- Mathematical computations often necessitate floating point operations. These must be explicitly enabled within STM32CubeIDE by navigating to Project > Properties > C/C++ Build > Settings. In the Tool Settings tab, select MCU/MPU Settings and mark “Use float with printf from newlib-nano (-u printf_float).”

Additionally, trigonometric functions are standard in nearly all compilers. The embedded Solar Tracker code developed by the National Renewable Energy Laboratory demonstrates these functions effectively. By entering specific coordinates and the current date, the program calculates solar position in the sky. This code can be adapted to control movable solar arrays, optimizing sunlight exposure and maximizing power generation.

Google Earth provides accurate GPS data—simply position the cursor on your location to view latitude, longitude, and elevation at the screen’s bottom-right. For information about the National Renewable Energy Laboratory site, enter the following coordinates into Google Earth: 39° 44' 32.9136" -105° 10' 42.96". Google Maps is also recommended for similar tasks.

Further details regarding the Solar Tracker code are available at https://midcdmz.nrel.gov/spa.

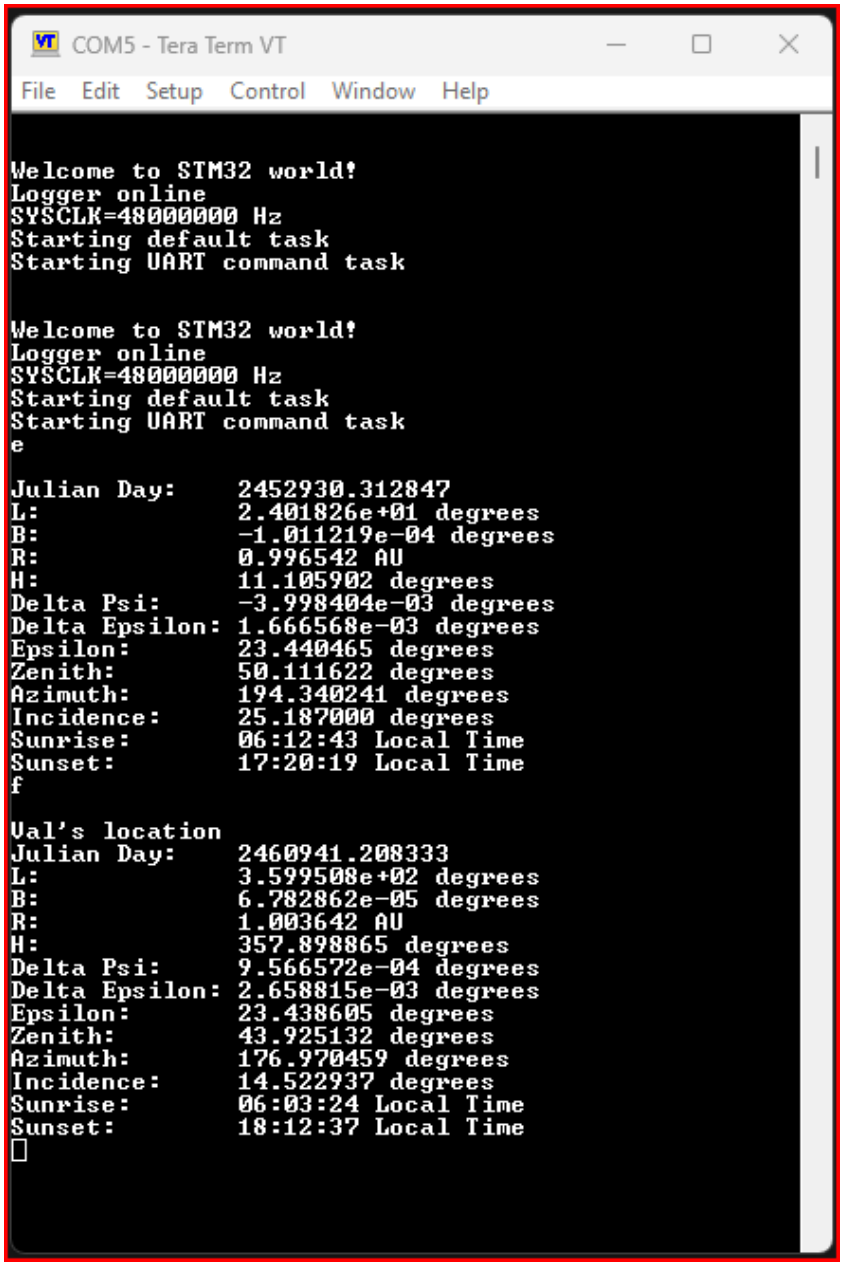
Local average atmospheric pressure and annual temperature statistics can be sourced through Google.

To convert between Degrees Minutes Seconds and Decimal Degrees, utilize the calculator at https://www.fcc.gov/media/radio/dms-decimal.

For a comprehensive list of time offsets by U.S. state and territory, visit https://*en.wikipedia.org/wiki/List_of_time_offsets_by_U.S.state_and_territory*.

This project incorporates additional files, including log.h, spa.h, log.c, spa.c, and spa_tester.c.

The Spa code, developed by the National Renewable Energy Laboratory, generates a predefined set of values if computations are completed successfully. Users may input their own coordinates into `spa_tester.c` to specify a particular location. To initiate the calculation and view the results, enter the command `e` followed by the Enter key for the original NREL location, or input `f` followed by Enter for a custom location. The output will be displayed as follows:



NOTES

Thread-safe functions are implemented in the files log.h and log.c. The following are examples:

```
log_puts(10, "Logger online\r\n");
log_printf(10, "SYSCLK=%lu Hz\r\n", HAL_RCC_GetSysClockFreq());
```

What waitMs is for

waitMs is the maximum time (milliseconds) the call will wait to:

1. Get a free buffer from the free-pool queue, and
2. Enqueue the message pointer into the ready-to-send queue.

If the pool is temporarily exhausted or the ready queue is full, the call will block up to waitMs trying again. If it still can't proceed, it returns pdFALSE (message dropped). This gives you back-pressure control per call.

- waitMs = 0 → non-blocking; drop if it can't proceed immediately.
- waitMs = 10 (or any finite) → wait a bit under load; good default.
- waitMs = portMAX_DELAY → block indefinitely (not for ISRs).

Choosing waitMs (rule of thumb)

- Control loops / tight deadlines: waitMs = 0 (never let logging delay control).
- General tasks: waitMs = 5–20 ms (brief tolerance to smooth bursts).
- Critical one-off diagnostics: waitMs = portMAX_DELAY (only if it's OK to block this task).
- ISRs: ignore; ISR API doesn't take waitMs. It never blocks.

The Log_Init function, invoked from main, configures the message queue of pointers and initiates a dedicated FreeRTOS task responsible for UART2 output. This task retrieves message pointers from the queue when pending messages are present. Other tasks submit message pointers to the queue via the log_puts or log_printf function calls.