A Study of Relative Humidity and Temperature on Sensor Modules for the Barrel Timing Layer of CMS

Christian Guinto-Brody

May 10, 2024

1 Introduction[1]

The Large Hadron Collider (LHC) will enter its High-Luminosity era in 2028, during which it will begin collecting 20 times more data than it is currently collecting. Increasing the LHC's luminosity, which measures how many collisions occur with each beam crossing, will increase the chances that rare processes are observed, providing researchers insight into undiscovered physics. To achieve these goals, a new subsystem of the CMS experiment, the Minimum Ionizing Particle Timing Detector (MTD), is being constructed to measure the time of arrival of particles with an unprecedented precision of 30 picoseconds. Increasing the precision with which particles are detected will allow researchers at CMS to keep track of the hundreds of more proton-proton collisions that will occur with every LHC beam crossing in the High-Luminosity era.

A key component of the MTD is the Barrel Timing Layer (BTL), which comprises many sensor modules that must be built and cured under just the right conditions for optimal light output. Conducting studies on the BTL sensor modules by testing the sensor modules' light output when built at various application pressures, room temperatures, and curing times will determine the optimal parameters for their construction. In addition, measuring the humidity of the assembly room will determine how it fluctuates and how those fluctuations affect the light yield of the sensor modules. By conducting these analyses, standardization in the production of sensor modules will be achieved, leading to precise and accurate light yield. This standardization will prime the MTD to work at maximum capacity for as long as possible, helping the LHC achieve its luminosity goals so new physics can be discovered.

2 The Barrel Timing Layer

The BTL will comprise sensor modules, which are composed of 16-bar LYSO scintillating crystals and two silicon photomultipliers (SiPMs). When collisions occur in the LHC, new particles will emanate from the collision point, interacting with the LYSO crystals of the BTL. The scintillation light will be collected by the SiPMs, creating an analog electrical signal. A circuit will then assign a time stamp to each particle with a resolution on the time of arrival of 30 picoseconds.

Currently, sensor module production consists of gluing two SiPMs to LYSO crystals with DOWSIL 3145 RTV MIL-A-46146 Adhesive/Sealant. The glue is applied using an applicator, which provides constant pressure to ensure that a uniform glue layer is produced. The glue then dries before the sensor module is placed in a climate-controlled box, where it cures. When the sensor modules have been cured, they are then connected in pairs by

[1] I would like to thank Chris Neu and Bryan Cardwell for their guidance in conducting this research and Maria, Jose, Reshma Menon, and Zhenyu Wu for supporting me, helping me with technical questions, and teaching me about the project. I would also like to thank Thomas Andersen for helping me set up the Arduino and Hayden Hollenback for telling me about WTTR.

TOFHiR readout electronics to form detector modules.[2] Twelve of these detector modules are then assembled to form readout units, and six of these readout units are assembled to form a tray. Figure 1 shows a model made by Zhenyu Wu of a tray. The BTL will be composed of 72 trays, meaning it will contain 10,368 sensor modules.

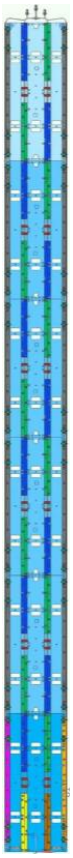
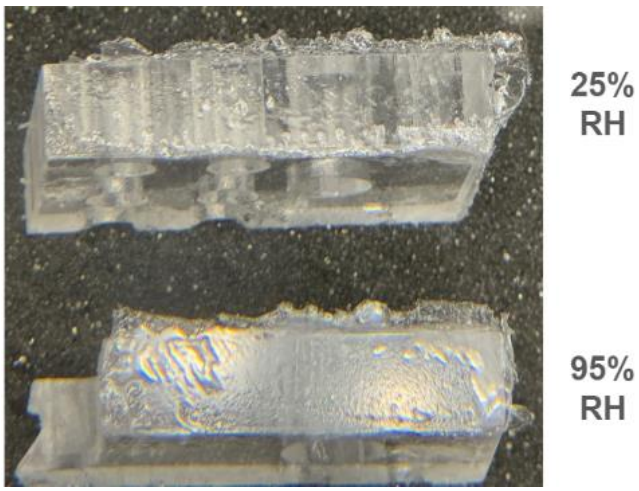


Figure 1: Model of one of the trays that will compose the BTL. Each tray is made of six readout units, which can be seen in the model as six evenly sized sections. The readout units are connected by long electronics that will detect and transmit information about the detected particles along the detector.

The electronics and sensor modules in the BTL must be kept at very low temperatures to ensure accurate light yield, so a cooling system running the length of the detector will be installed that will minimize noise from the SiPMs. Slight changes in the humidity of the room when curing the sensor modules could lead to deformities such as cracking and air bubbles in the glue, decreasing the lifetimes of the sensor modules and resulting in inefficient light yield and variations between them. These variations could also lead to unwanted phenomena such as crosstalk, which occurs when light enters one LYSO crystal and exits another, creating confusion about where the collision occurred.

A study conducted by researchers at Caltech analyzed the curing of sensor modules using similar silicon adhesive as is being used for constructing the BTL. They produced three mock sensor modules (made to mimic a sensor module but with plexiglass instead of SiPMs) using the glue and cured them in sealed containers containing different environments: one with low relative humidity (~20%), one with moderate relative humidity (~40%), and one with high relative humidity (~95%). After curing them for three and a half days, they removed them from their containers and observed how the curing went. Figure 2 displays their observations. The sensor module cured in 20% relative humidity exhibited brittleness and lots of air bubbles, neither of which is good for effective operation. The sensor module cured in 95% relative humidity, on the other hand, did not exhibit these aberrations, but its glue did not completely set, which is also not good for effective operation. The sensor module cured in 40% relative humidity showed the most promise, its glue having completely set and possessing no air bubbles or signs of brittleness. The researchers concluded that both very low and very high humidities were detrimental to the curing of sensor modules, and that the optimal range for curing was between 40% and 60% relative humidity.



[2] TOFHiR stands for Time-of-Flight High Rate. These electronics are specifically designed to achieve the unprecedentedly fast resolution promised by the MTD.

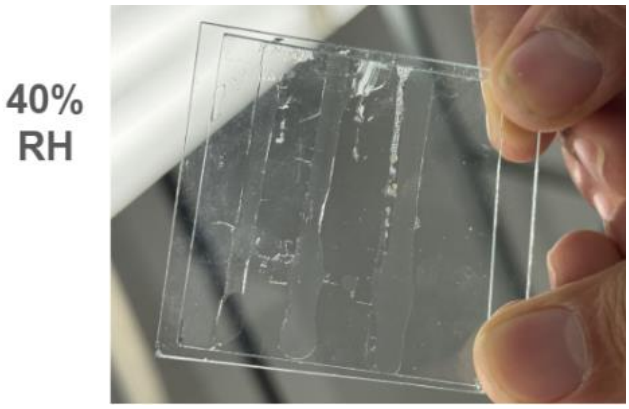


Figure 2: Closeups of sensor modules from Caltech study. Air bubbles and brittleness are visible in the glue cured in 20% relative humidity, while wetness is visible in the glue cured in 95% relative humidity. The sensor module cured in 40% relative humidity has none of these aberrations, indicating that it was cured in an environment with satisfactory humidity.

We will be building part of the BTL in the High-Energy Physics Building at the University of Virginia. Little information is known how the relative humidity in our assembly room fluctuates and how those fluctuations will affect the curing of the sensor module. To address these ideas, we set up an Arduino circuit that contains a sensor that will measure the humidity and temperature of the environment in which it is located. Using this data, we determined how these parameters changed in the potential assembly room. However, first it is important to understand the humidity and temperature and how they play a role in the function of SiPMs.

3 Relative Humidity, Temperature, and Other Meteorological Parameters

Meteorologists often distinguish between a few related types of humidity, all of which are defined by the amount of water vapor present in the air. The type of humidity that is most relevant to the construction of the BTL is relative humidity, which is the percentage of water vapor contained in the air relative to the maximum amount of water vapor that the air can hold at a given temperature.[3] Air can hold more water vapor at warmer temperatures, so if the amount of water vapor in the air stays constant, the relative humidity will be lower for hotter temperatures than for colder temperatures. Absolute humidity, on the other hand, is simply the amount of water vapor contained in the air regardless of the temperature and is measured in grams per cubic meter.

Meteorologists also distinguish between different types of temperatures, the most important for construction of the BTL being dry-bulb temperature—the temperature of the air measured by a thermometer freely exposed to it—and dewpoint temperature—the temperature at which the relative humidity of the air becomes 100% (also known as saturated) and water vapor starts to condense out of it.[4] Relative humidity can be defined in terms of these two temperatures by the relationship

$$t_d = \frac{B_1\left[\ln\left(\frac{\mathrm{RH}}{100}\right)+\frac{A_1 t}{B_1+t}\right]}{A_1-\ln\left(\frac{\mathrm{RH}}{100}\right)-\frac{A_1 t}{B_1+t}} \qquad (1)$$

where t_d and t are respectively the dewpoint and dry-bulb temperatures in ℃, RH is the relative humidity as a percentage, and $A_1 = 17.625$ and $B_1 = 243.04$℃ are empirical coefficients.

Figure 3 displays the relationship between t_d and RH based on (1) for various t. Each graph follows a logarithmic trajectory with a steep asymptote at $\mathrm{RH} = 0$, where the dewpoint temperature decreases rapidly to infinity. The graphs show that at a constant temperature, decreasing the relative humidity decreases the dewpoint temperature, and as the relative humidities approaches the low-end range, the dewpoint temperatures will become very cold very quickly. This is particularly important when operating sensor modules. If the temperature at which the module is located is held constant, the relative humidity of the air must be kept very low to keep the dewpoint temperature low and prevent condensation from forming on the sensor module, which could damage the electronics and result in inaccurate light yield. However, if the relative humidity in this range is increased very slightly, the dewpoint temperature will increase dramatically. Keeping the relative humidity not only low but constant will ensure that the dewpoint temperature stays at a manageable value that allows the sensor module to function.

[3] Temperature is defined as the measure of the hotness or coldness an object, the consequence of the speed at which molecules in that object are moving or vibrating.

[4] In usual conversation, when people refer to temperature, they are referring to the dry-bulb temperature. This report will continue this convention from this point forth.

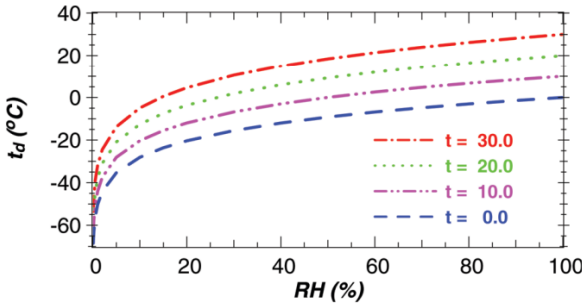


Figure 3: Dewpoint temperature as a function of relative humidity for various temperatures. At low relative humidities, the dewpoint temperature is also low but varies quickly, suggesting that the relative humidity must be kept constant and as low as possible to prevent the dewpoint temperature from increasing too much.

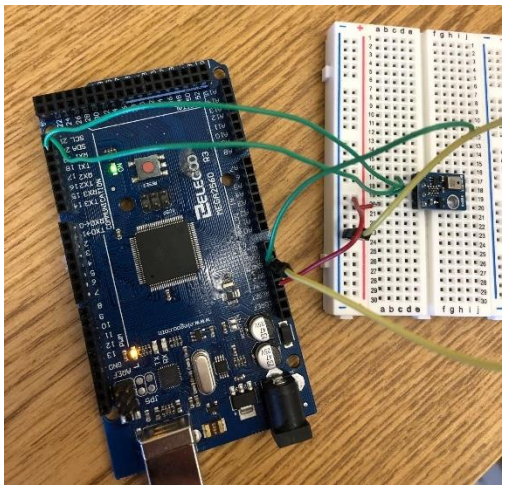


Figure 4: Circuit containing Arduino board and AHT10 humidity and temperature sensor. The Arduino board is in the foreground while the sensor is located on the breadboard to the right.

4 Collecting Data with Arduino and Humidity and Temperature Sensor

Figure 4 shows a photo of the circuit used to operate the humidity and temperature sensor. An Elegoo MEGA2560 R3 Arduino board was powered by a PC and thus gave power to a fully calibrated ASAIR AHT10 humidity and temperature sensor. The sensor came with a program that allows the user to set various parameters about the operation of the sensor (i.e. frequency of data collection, setup up code). When the code is run, any messages and data outputted by the Arduino are sent to a COM port where the PC can collect them. When data from the Arduino is sent to the COM port, Python scripts can access the serial port to collect the data, which can be written to data files and plotted.

We set up the Arduino and sensor such that it measured the relative humidity and temperature every second and sent those values to the COM port one after the other.[5] We then wrote two Python scripts that collected this data and plotted them.

One script, sensorData.py, uses Python's serial module to decode and sort the signals coming from the Arduino. This script also fetches data from WTTR, a weather-collection program that can provide various meteorological data from most locations around the world. These data were plotted alongside the data collected by the sensor to see how the humidity and temperature in the assembly room compared with those of Charlottesville. sensorData.py takes two arguments from the command line: the name of the data file to which the measurements from the sensor should be written and the name of the data file to which the measurements from WTTR should be written (if no file for WTTR data is specified, a default file is used).

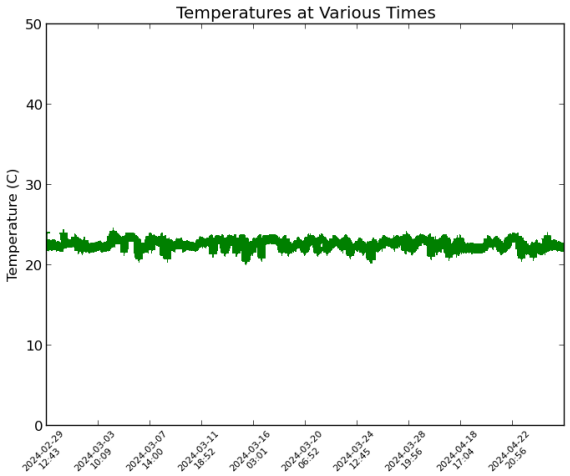
Once run, sensorData.py will collect data and put it in data files until it is forced to stop, either by exiting out of the program or by clicking the "space" bar. When the

$$[\text{bytes}] = \left[\frac{\text{bytes}}{\text{point}}\right] \cdot \left[\frac{\text{points}}{\text{hour}}\right] \cdot \left[\frac{\text{hours}}{\text{day}}\right] \cdot [\text{days}] \qquad (2)$$

If we collected one measurement per second (i.e. 3,600 points per hour) for the duration of the BTL assembly (~1.5 years, or ~538 days), the resulting data file would be ~1.5 gigabytes large. The directory holding our data still has several gigabytes of free storage, indicating that collecting data at a rate of one measurement per second is a safe rate that will not use all the PC's storage.

[5] The decision to collect measurements every second came after determining how much storage a data file occupied as a function of the number of measurements it held. We were limited on storage and did not want to use all the PC's storage with data. The relationship between these variables was almost linear, with each point holding about 32 bytes of storage. Since measurements were collected periodically, and since the data files grew with time, the number of bytes held by a data file was given by

latter happens, the data are plotted by importing the second script, dataReader.py. sensorData.py will feed dataReader.py the two data files that respectively contain the sensor and WTTR data and plot them, humidities going on one plot and temperatures going on another. dataReader.py can also be used independently of sensorData.py; in this case, dataReader.py takes two existing data files from the command line as well as a number that indicates that it is being used independently. A third script, otherDataReader.py, is the same as dataReader.py, except that it only plots data collected by the sensor.



5 Plots of Sensor and WTTR Data

Figure 5 displays humidity and temperature data collected by the sensor in the assembly room over a period between February 29, 2024, at 12:43 pm to May 2, 2024, at 4:52 pm. We observed that the temperature in the room stayed roughly constant at about 22°C, which is to be expected since the building is temperature controlled. The humidity, however, fluctuated significantly, acquiring values as low as 15% and as high as almost 70%. On dry days in the winter, the humidity stayed very low—often below 20%—but when it rained, the humidity increased dramatically, indicating that the room was very susceptible to increases of moisture. We also observed that a relatively small percentage—only ~30%—of days saw humidities between 40% and 60%, the optimal humidity range for curing sensor modules as determined by Caltech's study.

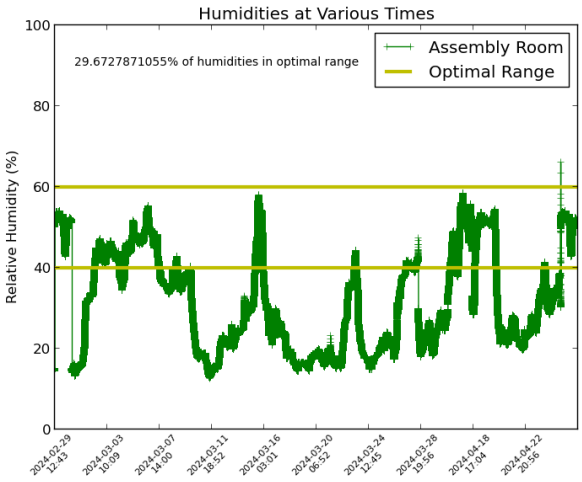


Figure 5: Humidities (top) and temperatures (bottom) of the assembly room as measured by the sensor between late February and early May. The temperature in the room stayed roughly constant while the humidity fluctuated frequently and significantly.

It is important to note that the data were collected sporadically; while the graph looks continuous, there are periods of time when no data was collected. The continuousness of the graph is a result of how the data file was created—each data file from each run of data collection was merged without filling in data for the dates where there was no data collection. However, there are enough data to suppose the trends in both humidity and temperature still hold: the former increased drastically when there was precipitation; the latter stayed roughly constant.

Figure 6 displays a smaller range as given in Figure 5, plotting humidity and temperature data only between April 30, 2024, at 2:09 pm to May 2, 2024, at 4:52 pm. These plots, however, also contain data collected by WTTR to compare with the sensor data, including precipitation data to determine whether rain increases the humidity in the assembly room. We observed that the temperature in the room stayed very constant but the temperature in Charlottesville fluctuated, decreasing during nighttime hours and increasing during daytime hours. This plot confirms that the temperature in the assembly room is not susceptible to changes in temperature outside. We also observed that the humidity was much higher than normal for these three days, despite it only raining for a little bit of time. The humidity also did not increase when it rained—in fact, it appeared to have decreased after the rain stopped. These occurrences are likely due to the weather getting substantially warmer for these past few days, independently causing the moisture in the outside air to increase, thus increasing the humidity in the assembly room.

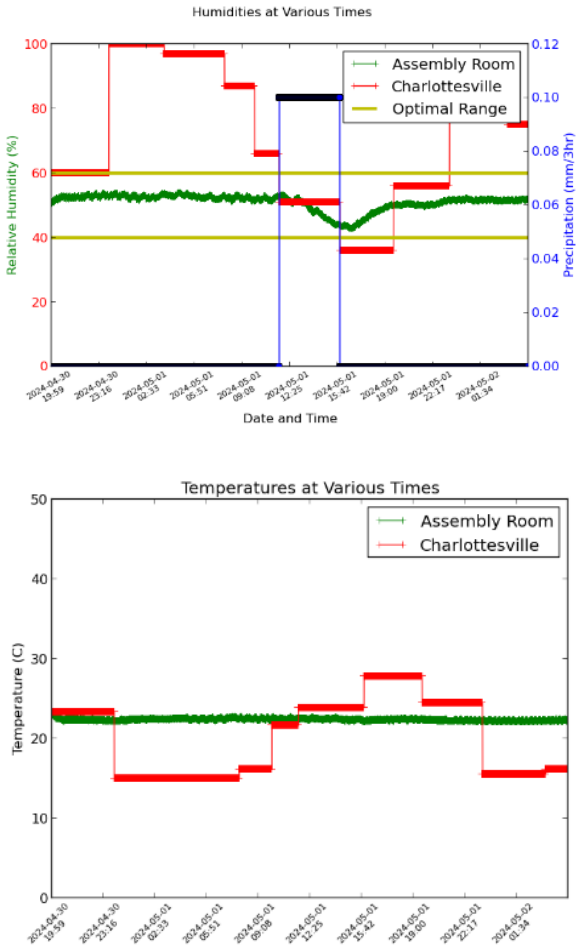


Figure 6: Humidities (top) and temperatures (bottom) from the sensor and WTTR between the afternoons of April 30, 2024, and May 2, 2024. The humidity and temperature in Charlottesville (shown in red) fluctuated much more than that measured by the sensor (shown in green). The former graph also plots the precipitation measured in Charlottesville to determine whether there exists a correlation between the humidity detected by the sensor and the amount of rain outside.

6 Discussion and Future Work

From this study, we concluded that the temperature in the assembly room does not fluctuate significantly enough to be a cause of concern; however, the humidity in the assembly room not only normally resides as a very low level, but it fluctuates drastically when it rains, suggesting that lots of moisture enters the room during these events. The humidity in the room is usually less than 20%, which, according to the results of the study done by Caltech, would cause cracking and air bubbles in the RTV glue of the sensor modules if we cured them in the room. Rain brings the humidity in the room to the optimal range; however, when it stops raining, the humidity drops back to its suboptimal level. In addition, frequent changes in the humidity could cause unwanted damage to the sensor modules, decreasing their life expectancy and light yield even more. To keep the humidity at a constant level and within the optimal range, we need to invest in a humidifier for both the room and the device in which we cure the sensor modules. Using a humidifier would prevent aberrations from forming in the glue when curing the sensor modules, helping produce modules with maximum light yield and life expectancy.

7 Conclusion

In four years, the LHC will enter its High-Luminosity era, where it will collect 20 times more data than it currently collects. The MTD will help the LHC achieve this goal by providing it with unparalleled timing resolution. A major component of the MTD is the BTL, which consists of thousands of sensor modules. These devices are very susceptible to changes in humidity and temperature and must be assembled in just the right conditions to function properly and effectively. We used an Arduino board and a humidity and temperature sensor to measure these quantities in the room in which we will be building sensor modules. Over the course of three or so months, we found that the temperature in the room stays roughly constant while the humidity increases dramatically when it rains, bringing it from a level that is too low for effective sensor-module production to the optimal range as determined by researchers at Caltech. To combat these issues, we must put a humidifier in the assembly room so that sudden changes in humidity do not occur and so that the room is typically at a humidity level that prevents aberrations from forming in the RTV glue of the sensor modules. By conducting this study, we determined that the assembly room is not quite suited for producing sensor modules. Knowing this will prevent us from making too many faulty sensor modules, helping us speed up building the BTL efficiently and effectively.