

**MODULE 3-4**

# REAL TIME CLOCK

# Introduction – Real Time Clock

An RTC is an electronic time keeping device with its own battery. It's often used in other devices such as a data logger.

## **Purpose:**

- Timestamping
- Event management (helps with power saving)

## **Features**

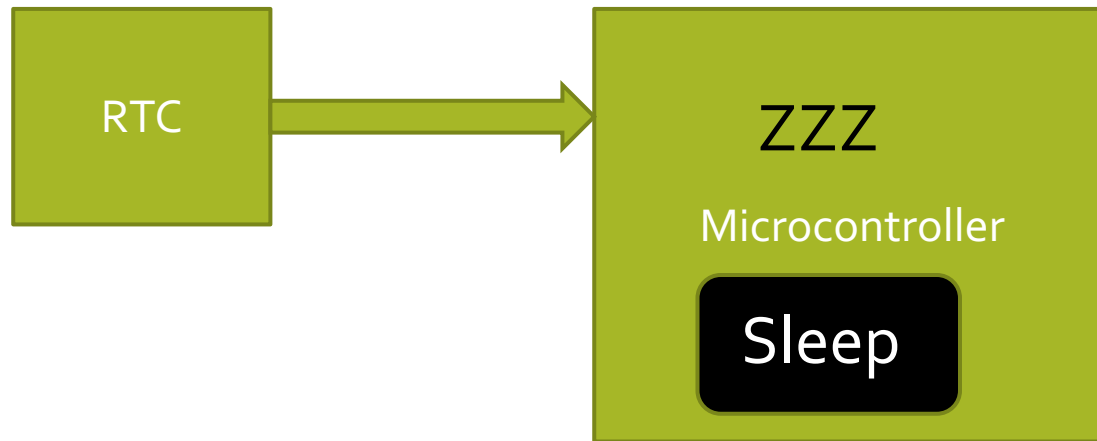
- Has its own battery
- Keeps time even when the main device is powered off or the batteries are dead

# Introduction – Real Time Clock

Event management (helps with power saving)

- Alarm
- Timer

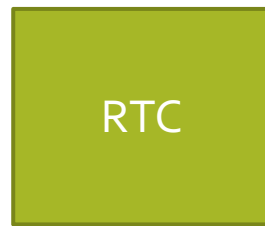
# Real Time Clock - Alarm/Timer



# Real Time Clock - Alarm/Timer



# Real Time Clock - Alarm/Timer

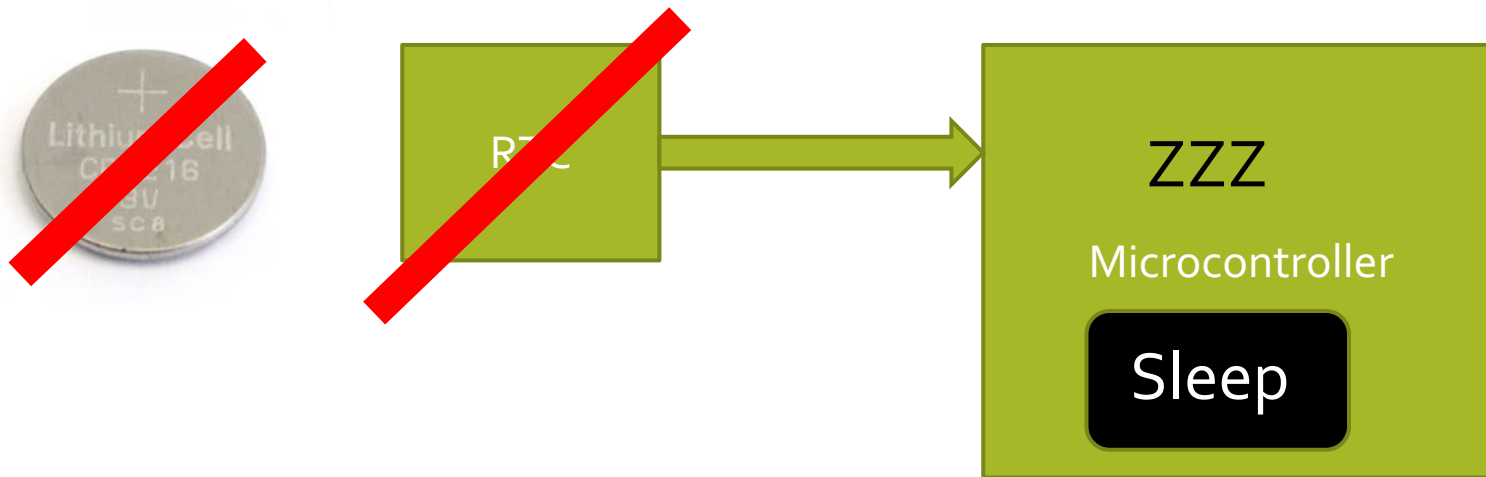


Read Sensors

Read Date/Time

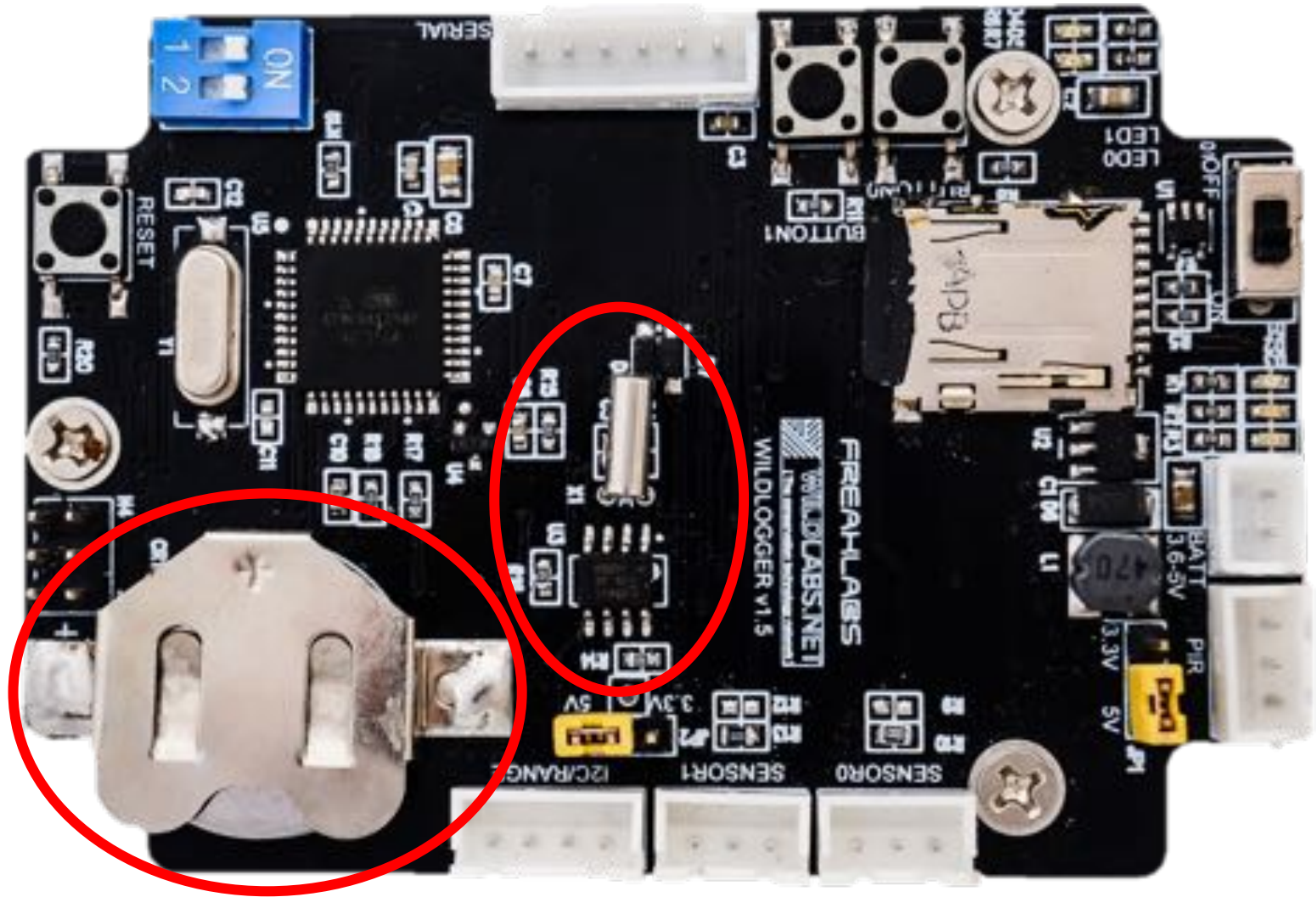
Write to SD

# Real Time Clock - Alarm/Timer



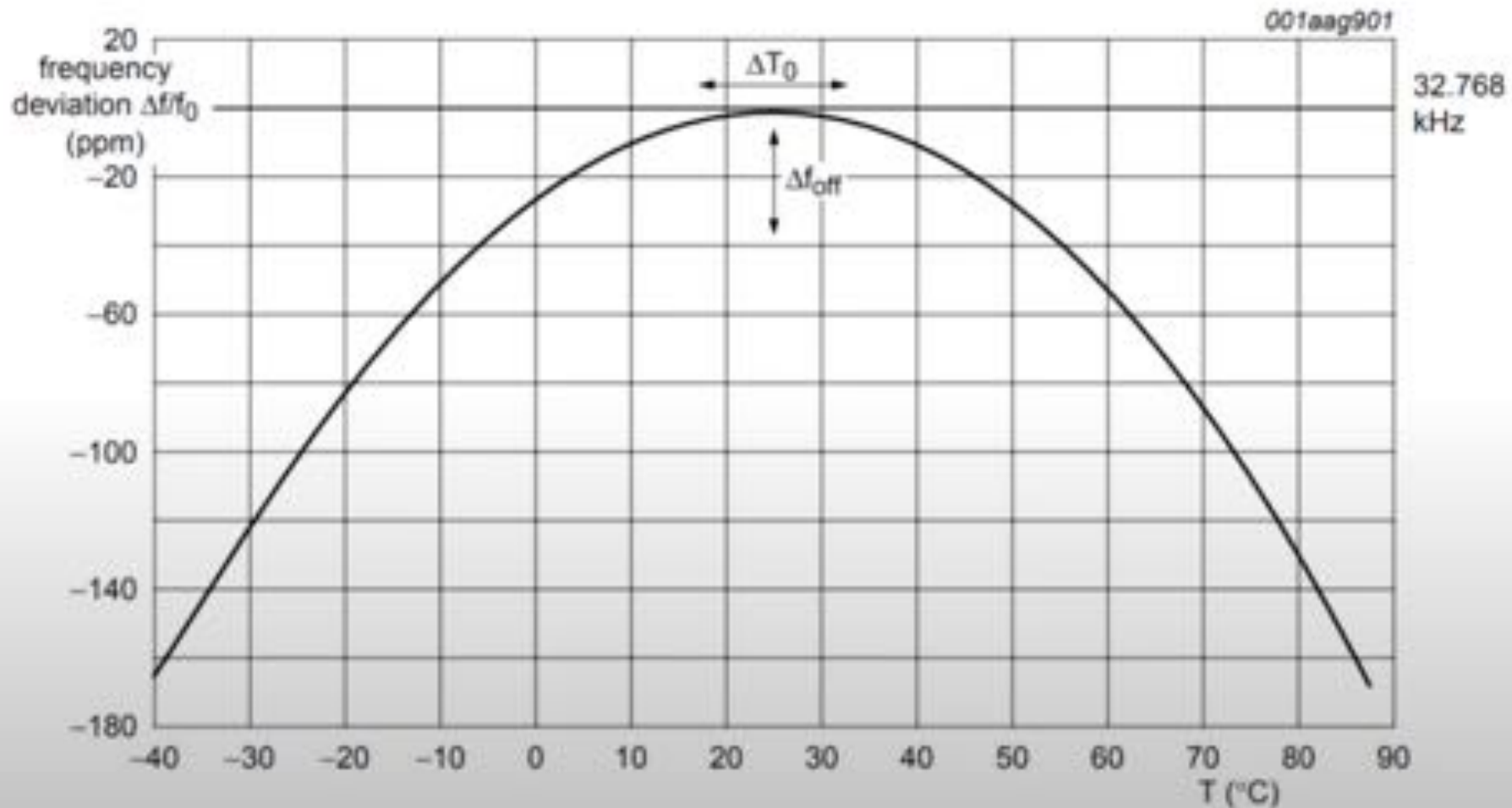
Catastrophic Failure!

# WildLogger – Real Time Clock

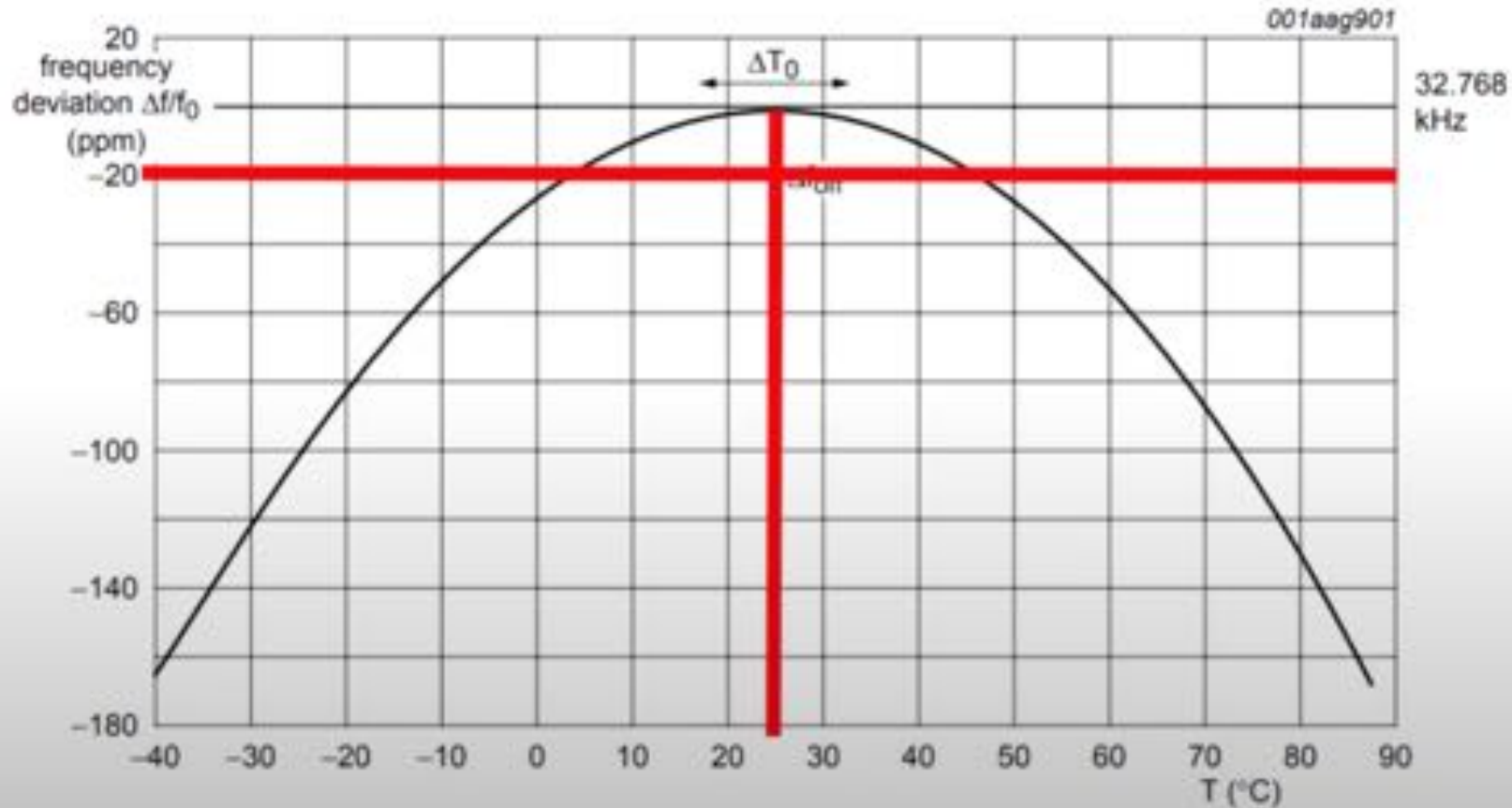




# Real Time Clock - Accuracy



# Real Time Clock - Accuracy

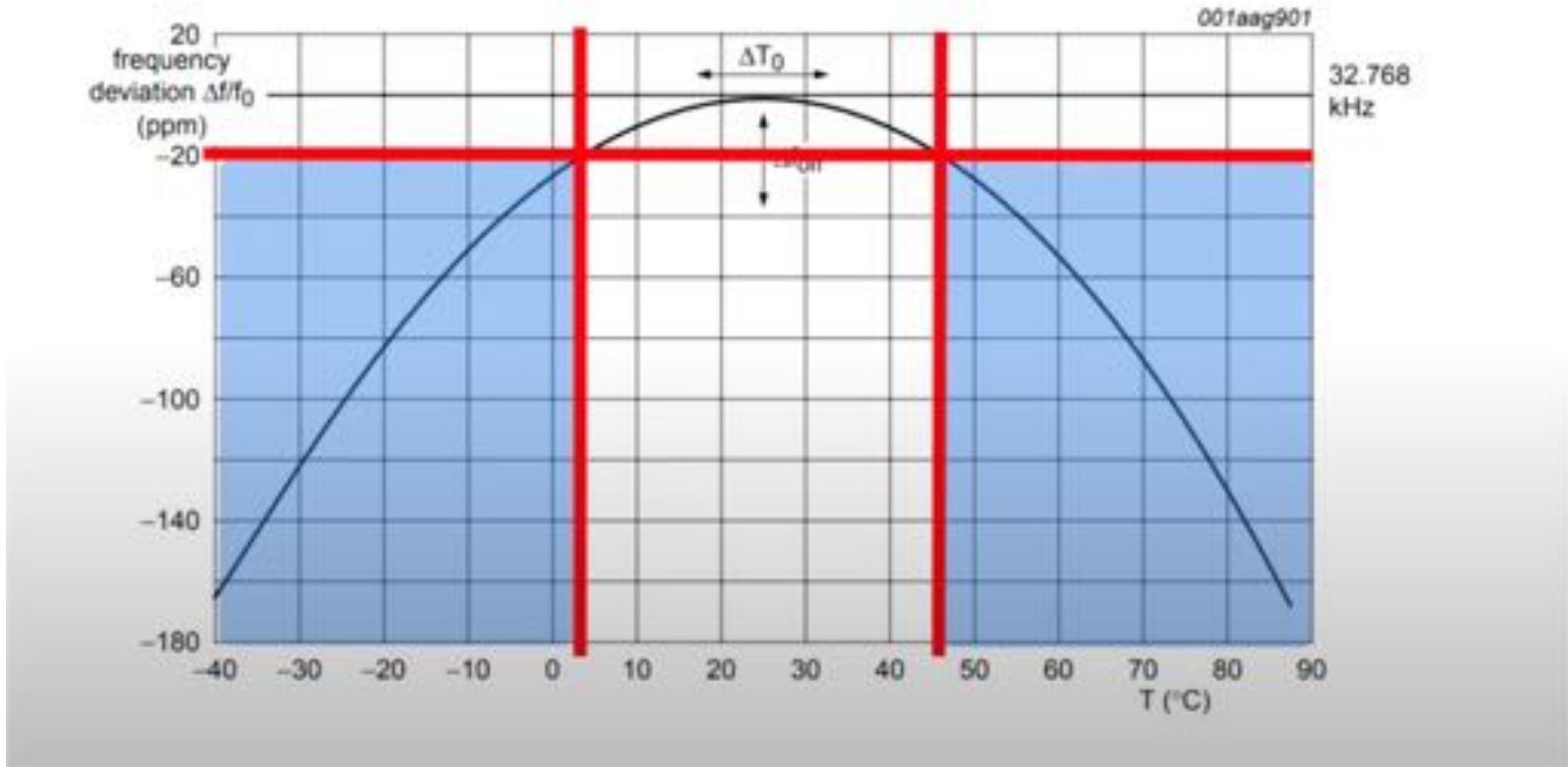


10 ppm = loses 0.5 min per month (approx)

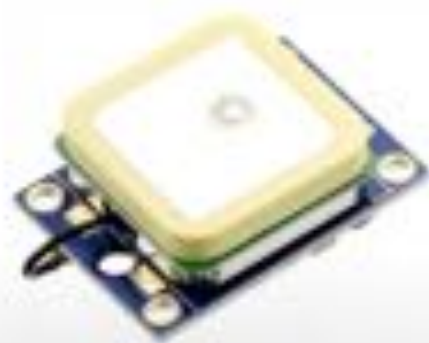
20 ppm = loses 1 min per month (approx)

40 ppm = loses 2 min per month (approx)

# Real Time Clock - Accuracy



# Real Time Clock Accuracy





# Module 3-4

## Real Time Clock

### Lab 4a

## What Time Is It?

# Lab 4a – What Time Is It?

## **Goal:**

- Install real time clock library
- Set time and date
- Read time and date

# Lab 4a – What Time Is It?

## Why do I need to know this?

- Timekeeping will be critical to the WildLogger functionality
- Reading time will be used to timestamp data
- Timekeeping useful in many other applications, esp. low power ones



# Lab 4a – What Time Is It?

## What do I need to know?

- Install PCF8563 real time clock library manually
- **Rtc\_Pcf8563 rtc**
  - Creates a real time clock (RTC) object
- **rtc.setDate(day, weekday, month, century, year);**
  - We can set century to 0 for 20xx
- **rtc.setTime(hour, minute, second)**
  - Note: hour is in 24-hour format



# Lab 4a – What Time Is It?

- **getWeekday(), getDay(), getMonth(), getYear()**
  - Gets the numerical (integer) value for each
- **getSecond(), getMinute(), getHour()**
  - Gets the numerical (integer) value for each
- **char \*rtc.formatDate(format)**
  - Returns the date in a pre-formatted string mm/dd/yyyy
  - Can change format also:
    - RTCC\_DATE\_CZ - dd.mm.yyyy
    - **RTCC\_DATE\_ASIA - yyyy-mm-dd (what we'll use)**
    - RTCC\_DATE\_WORLD - dd-mm-yyyy
    - RTCC\_DATE\_US – mm/dd/yyyy (default)
- **char \*rtc.formatTime()**
  - Returns the time in a pre-formatted string hh:mm:ss

# Lab 4a – What Time Is It?

- `rtc.setDate(day, weekday, month, century, year);`

Week Day	Sunday = 0 Monday = 1 Tuesday = 2 Wednesday = 3 Thursday = 4 Friday = 5 Saturday = 6
Day of the month	As per calendar day (1 – 31)
Month of the year	As per calendar month (1-12)
Century	0 = 2000 1 = 1900
Year	Last two digits of calendar year (0-99)

# Lab 4a – What Time Is It?

- `rtc.setTime(hour, minute, second)`

*Hours (oo), minutes(oo), seconds(oo)*

*eg. 12:34:21*

*24 hours*

12am = 00

1am = 01

2am = 02

3 am = 03 etc

12 noon = 12

1pm = 13

2pm = 14

3pm = 15 etc

# Manually Installing an Arduino Library

[https://github.com/elpaso/Rtc\\_Pcf8563](https://github.com/elpaso/Rtc_Pcf8563)

Search or jump to... Pull requests Issues Marketplace Explore

elpaso / Rtc\_Pcf8563 Watch 3 Star 19 Fork 19

Code Issues 2 Pull requests 2 Actions Projects Wiki Security Insights

master 1 branch 0 tags

Go to file Add file Code

elpaso Merge pull request #17 from filprojek/master

- examples Add Serial.begin() to setup
- README.rst Readme
- Rtc\_Pcf8563.cpp add 12h format and czech date fc
- Rtc\_Pcf8563.h add 12h format and czech date fc
- keywords.txt Use correct field separator in key
- library.json @platformio Library Registry manifest file 5 years ago

Clone  
HTTPS SSH GitHub CLI  
[https://github.com/elpaso/Rtc\\_Pcf8563](https://github.com/elpaso/Rtc_Pcf8563)  
Use Git or checkout with SVN using the web URL.

Open with GitHub Desktop

Download ZIP

About  
Arduino Pcf8563 RTC library. A fixed/expanded version of the original Rtc\_Pcf8563.  
Readme

Releases  
No releases published

Packages  
No packages published

Contributors 6

Languages

README.rst

This is my fixed/expanded version of the original Rtc\_Pcf8563 library for Arduino. All credits go to the original authors.

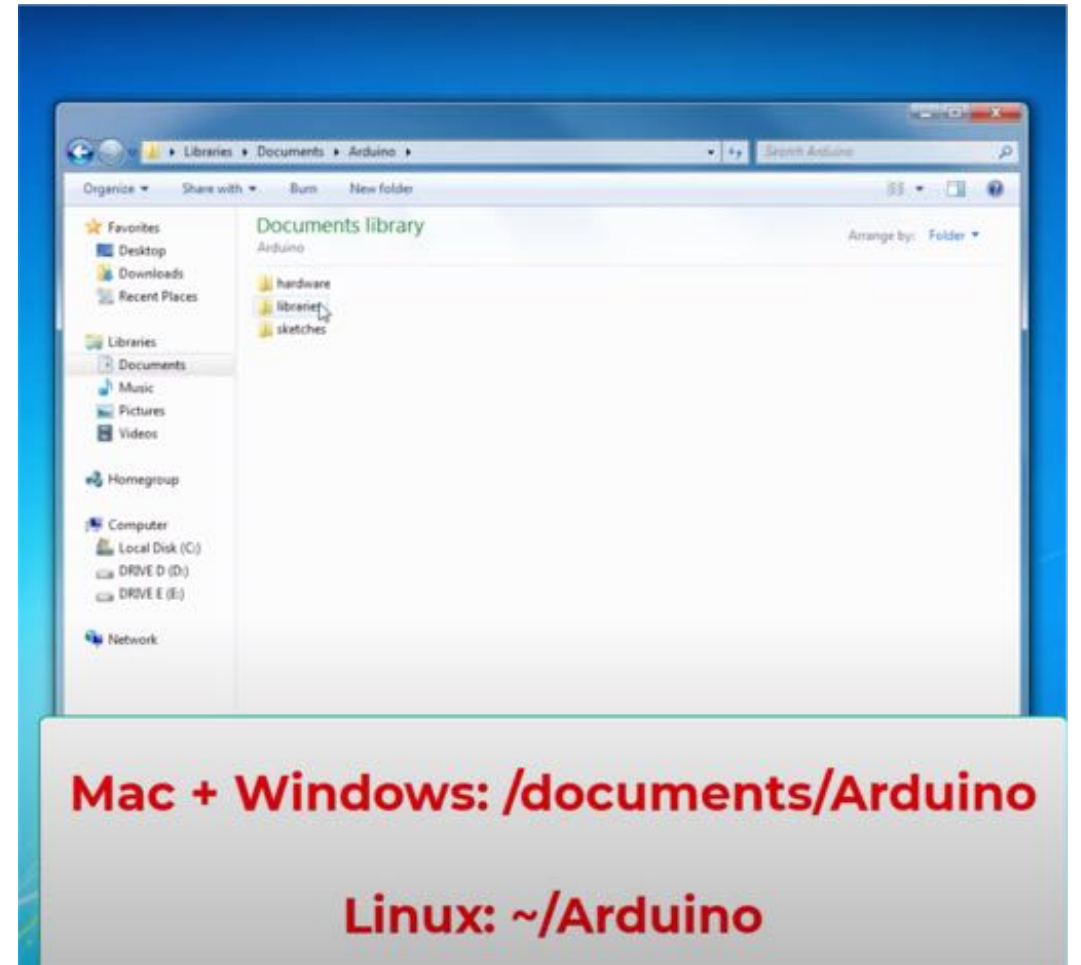
My version differs for

- a few more methods, mostly useful for debugging
- a single bug fix in RTCC\_ALARM\_AF

# Manually Installing an Arduino Library

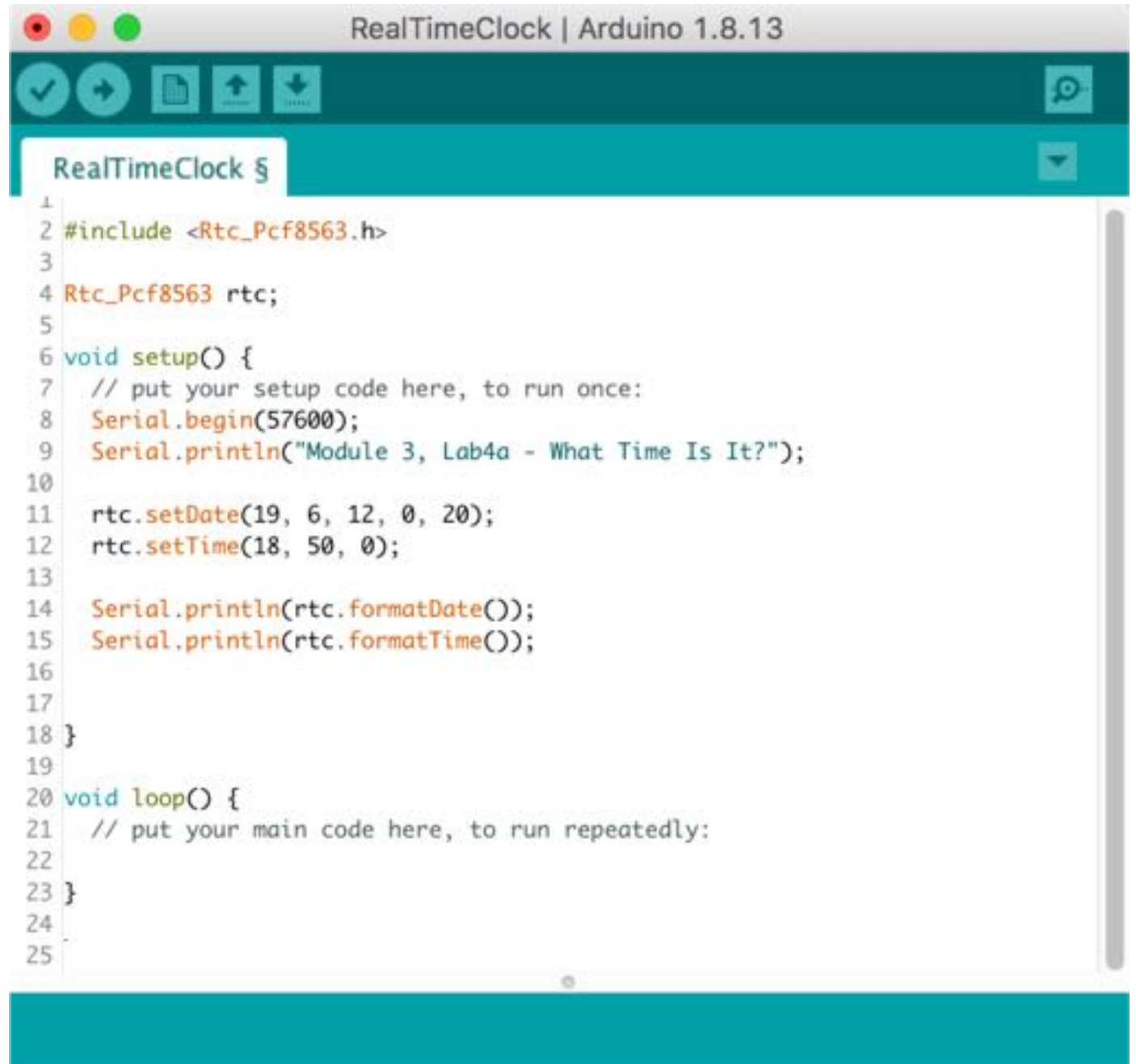
/Documents/Arduino/libraries

~ /Arduino/libraries



# Lab 4a

## Real Time Clock



```
RealTimeClock | Arduino 1.8.13
RealTimeClock §
1
2 #include <Rtc_Pcf8563.h>
3
4 Rtc_Pcf8563 rtc;
5
6 void setup() {
7   // put your setup code here, to run once:
8   Serial.begin(57600);
9   Serial.println("Module 3, Lab4a - What Time Is It?");
10
11   rtc.setDate(19, 6, 12, 0, 20);
12   rtc.setTime(18, 50, 0);
13
14   Serial.println(rtc.formatDate());
15   Serial.println(rtc.formatTime());
16
17
18 }
19
20 void loop() {
21   // put your main code here, to run repeatedly:
22
23 }
24
25
```



**Module 3-4**  
**Real time clock**

**Lab 4b**

**Command Line Time and Date**

# Lab 4b – Command Line Time

## **Goal:**

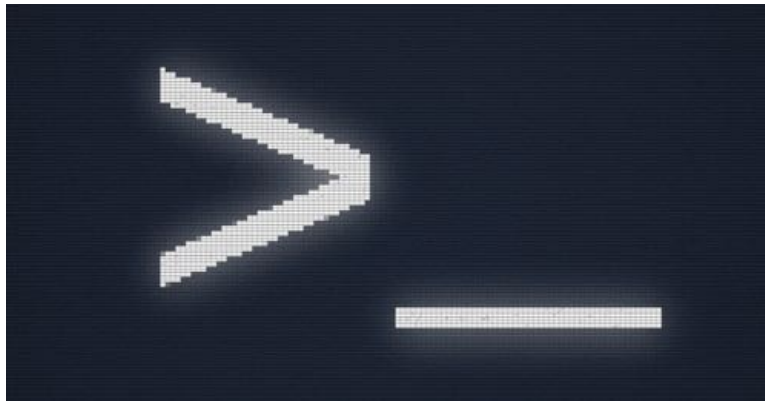
- Create a command that sets the time
- Create a command that sets the date
- Create a command that prints out the time
- Create a command that prints out the date



# Lab 4b – Command Line Time

## Why do I need to know this?

- Extremely useful to have a command to set time and date. Usually just done once before deployment
- Test out our time and date format as a timestamp



# Lab 4b – Command Line Time

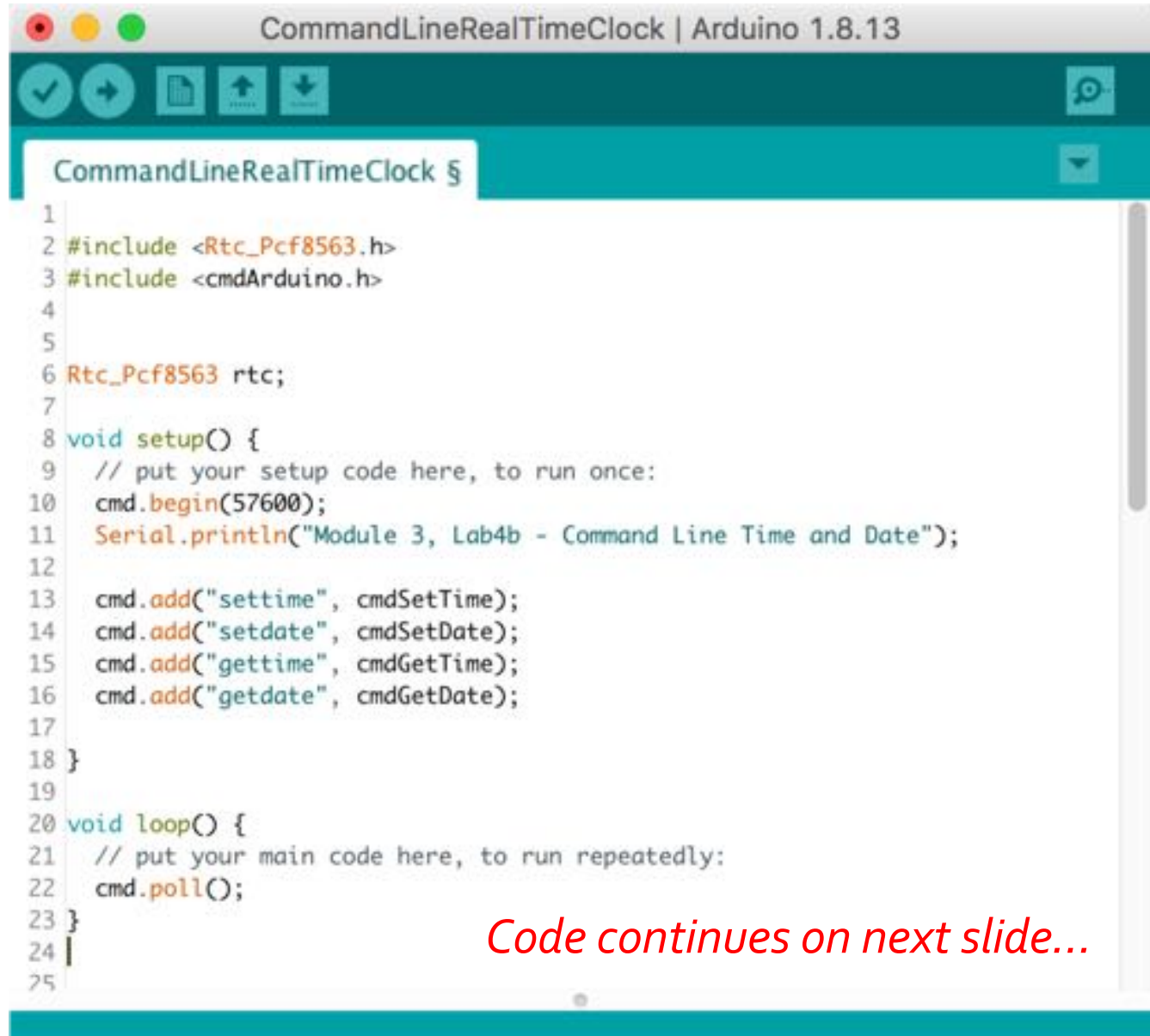
What do I need to know?



# Lab 4b

## Command Line Time and Date

*Code continues on next slide...*



```
CommandLineRealTimeClock | Arduino 1.8.13
CommandLineRealTimeClock §
1
2 #include <Rtc_Pcf8563.h>
3 #include <cmdArduino.h>
4
5
6 Rtc_Pcf8563 rtc;
7
8 void setup() {
9     // put your setup code here, to run once:
10    cmd.begin(57600);
11    Serial.println("Module 3, Lab4b - Command Line Time and Date");
12
13    cmd.add("settime", cmdSetTime);
14    cmd.add("setdate", cmdSetDate);
15    cmd.add("gettime", cmdGetTime);
16    cmd.add("getdate", cmdGetDate);
17
18 }
19
20 void loop() {
21     // put your main code here, to run repeatedly:
22     cmd.poll();
23 }
24 |
25
```

*Code continues on next slide...*

# Lab 4b

## Command Line Time and Date

*This code goes after the loop function...*



```
25 void cmdSetTime(int argCnt, char **args)
26 {
27   int hour = cmd.conv(args[1]);
28   int minute = cmd.conv(args[2]);
29   int second = cmd.conv(args[3]);
30
31   rtc.setTime(hour, minute, second);
32   Serial.println(rtc.formatTime());
33
34 }
35
36 void cmdSetDate(int argCnt, char **args)
37 {
38   int year = cmd.conv(args[1]);
39   int month = cmd.conv(args[2]);
40   int day = cmd.conv(args[3]);
41   int weekday = cmd.conv(args[4]);
42
43   rtc.setDate(day, weekday, month, 0, year);
44   Serial.println(rtc.formatDate());
45 }
46
47
48 void cmdGetTime(int argCnt, char **args)
49 {
50   Serial.println(rtc.formatTime());
51 }
52
53 void cmdGetDate(int argCnt, char **args)
54 {
55
56   Serial.println(rtc.formatDate());
57 }
58
en
```