

# **Smart Integrated Tile (S.I.T.)**

---

Project Update  
Sam, Ivy, Tony

# Motivation Why are we doing this?

- Scientists build 3D models to keep track of changes of coral reefs
- Depth information is needed...
- Current method is by manually measuring the depth: **hard, not accurate**
- A smart tile that automates this process: measure and display the depth under water. Divers with 3D camera will capture photos to get depth information.



# Requirements More details...

- Waterproof (~10 meters)
- Measure the depth and display
- Rechargeable battery
- Reprogrammable
- Ability to turn on/off
- Log and transfer the depth data
- Cheap (less than \$100) in order to be widely distributable to collaborators

# Challenges

## Waterproof is too hard...

- iPhone X (IP67 rated)
  - Half an hour
  - Up to 1 meter
- Our Device
  - 4 hours
  - 10 to 15 meters
  - Salt water



# Challenges    Waterproof means more

- No buttons. No plugs
    - Exposed cords and pins will erode in salt water
    - Waterproof plugs or buttons are not affordable
  - How to reprogram without plugs?
  - How to turn ON/OFF the device?
- 
- 
- 
- 
- 
- 
- 
- 
- **Solution:**
    - Use water-resistant plugs as a temporal solution
    - Wireless charging & wireless data transmission
    - Reed switch for turning ON/OFF
    - Pogo pins for reprogramming

# Challenges A clear display is important

- The tile should display the depth clearly enough **under all conditions / in camera**
- Condition under water: **sunlight, wind, wave**
- May have additional illumination

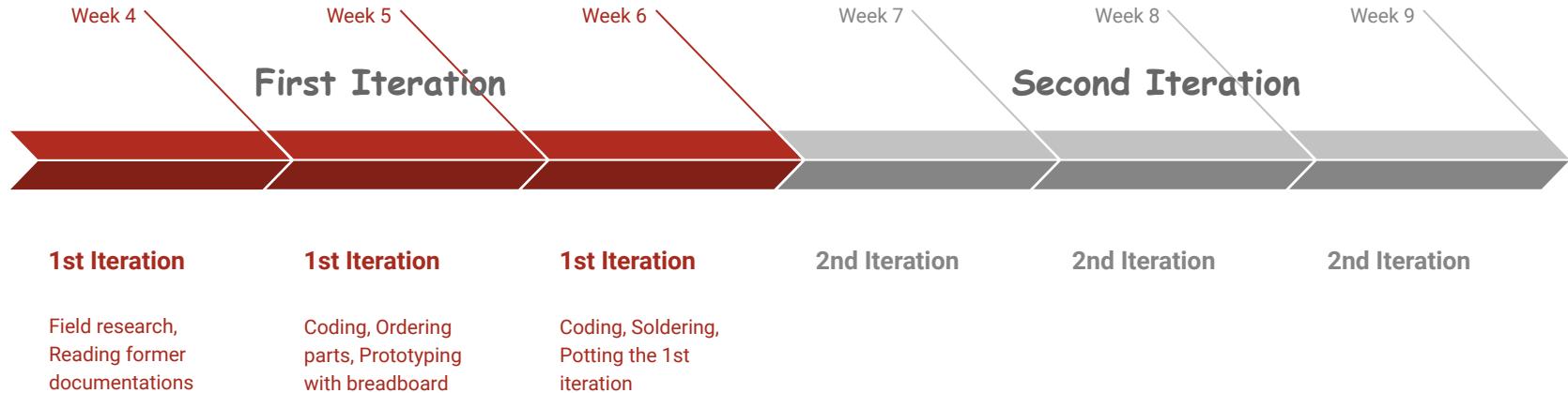


- **Solution:** Try and test different types of display module **LED, e-paper**



# Objectives

- Develop and test **two different prototypes**
- First iteration: displays and sensors
- Second iteration: data transmission, more waterproof



# Accomplishment

1st Iteration: a **fully integrated** device that could be tested underwater

- Measure the depth with depth sensor
- Two display modules
- Reed Switch to turn on/off
- Battery management and display battery status

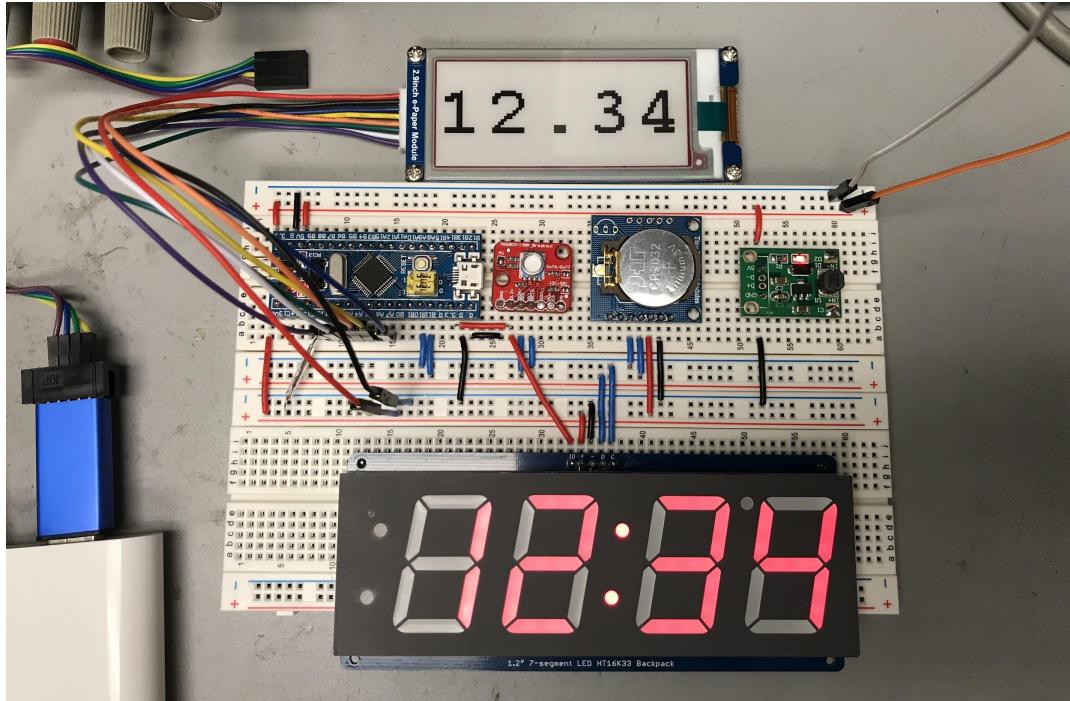


# Progression

- **Buying parts**
  - Breadboard
  - Coding
  - Soldering
  - Potting
  - Testing

# Progression

- Buying parts
- **Breadboard**
- Coding
- Soldering
- Potting
- Testing



# Progression

- Buying parts
  - Breadboard
  - **Coding**
  - Soldering
  - Potting
  - Testing

The screenshot shows the Keil MDK-ARM IDE interface. The left pane displays the project structure:

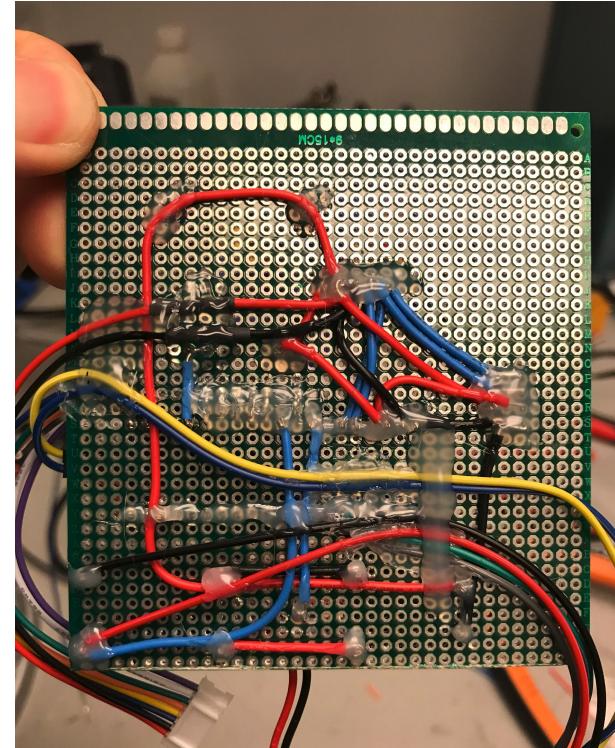
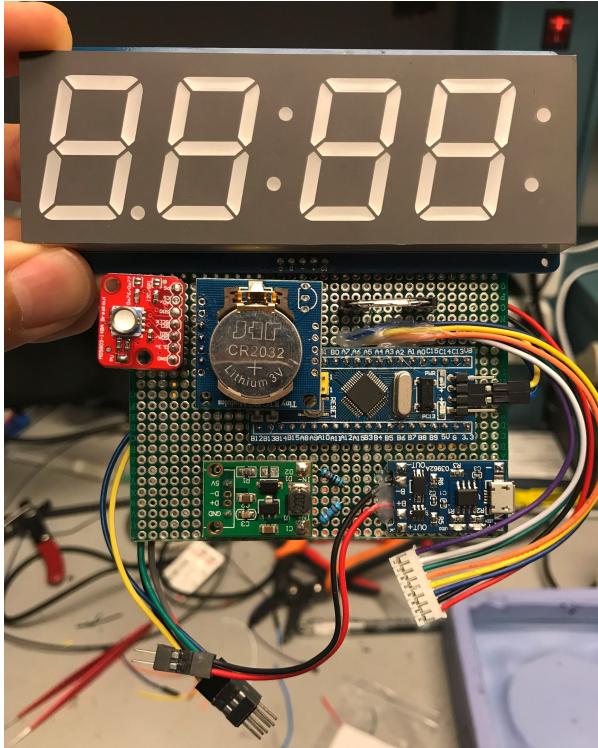
- Project: i2c-master-polling
- STM32F103C8
- Source:
  - delay.c
  - i2c.c
  - main.c
  - uart.c
  - ds1307.c
  - ms5803.c
  - led\_display.c
  - adc.c
  - spi.c
  - eink.c
  - wkup.c
- Include:
  - delay.h
  - i2c.h
  - uart.h
  - ds1307.h
  - ms5803.h
  - led\_display.h
  - adc.h
  - spi.h
  - eink.h
  - wkup.h
- Fonts:
  - fonts.h
  - font8.c
  - font24.c
- CMSIS
- Device

The right pane shows the content of the main.c file:

```
1 #include "stm32f10x.h"
2 #include "stm32f10x_rcc.h"
3 #include "stm32f10x_gpio.h"
4
5 #include "wkup.h"
6 #include "delay.h"
7 #include "usart.h"
8 #include "i2c.h"
9 #include "ms5803.h"
10 #include "ds1307.h"
11 #include "led_display.h"
12 #include "adc.h"
13 #include "eink.h"
14
15 // PA5: Tx
16 // PA10: Rx
17 // PB10: SCL
18 // PB11: SDA
19
20
21 void Device_Setup() {
22     // wake up if pressed more than 2s
23     DelayInit();
24     WKUP_Init();
25
26     // initialize peripherals
27     USART1_Init();
28     printf("#####\n");
29     printf("USART1 Initialized\n");
30     Eink_Init();
31     printf("Eink Display Initialized\n");
32     ADC1_Init();
33     printf("ADC1 Initialized\n");
34     ds1307_init();
35     printf("RTC Initialized\n");
36     ms5803_init(ADDRESS_HIGH);
37     printf("Depth Sensor Initialized\n");
38     LED_7_Seg_Init();
39     printf("LED Display Initialized\n");
40     printf("-----\n");
41     printf("All Peripherals Initialized\n");
42     printf("-----\n");
43
44     // register standby functions for peripherals
45     Register_Standby_Funcs(LED_7_Seg_Standby);
46     Register_Standby_Funcs(Eink_Standby);
47 }
48
49 static uint8_t welcome_flag = 1;
50
51 int main(void)
```

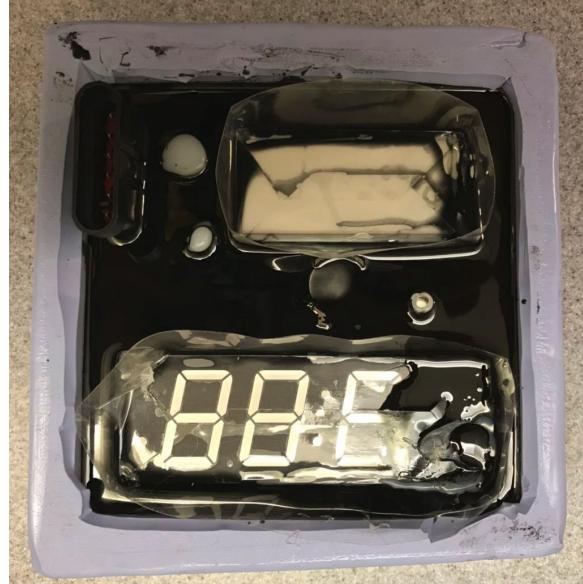
# Progression

- Buying parts
- Breadboard
- Coding
- **Soldering**
- Potting
- Testing



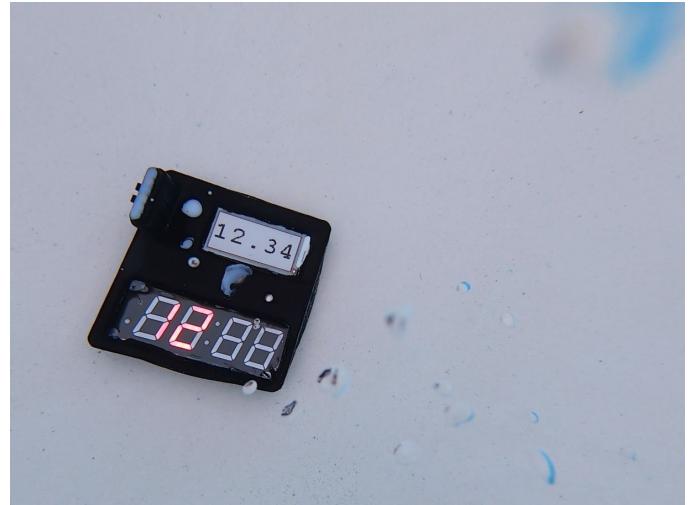
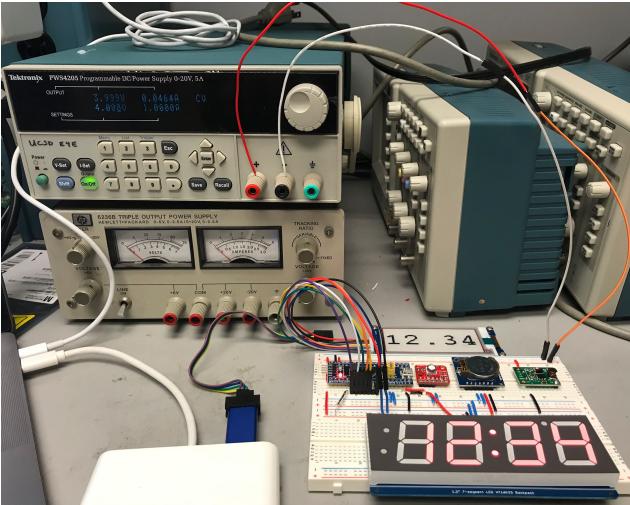
# Progression

- Buying parts
- Breadboard
- Coding
- Soldering
- **Potting**
- Testing

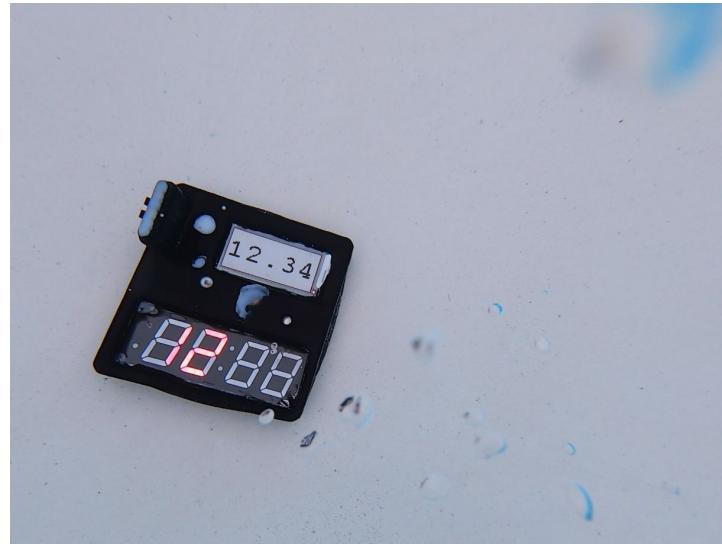


# Progression

- Buying parts
- Breadboard
- Coding
- Soldering
- Potting
- **Testing**



# Demo



# What to do next

## Second Prototype:

- Boards: PCB
- Sensors: depth, RTC
- Displays: E-ink, LED
- Charging: wireless
- Switch: reed switch for turning on/off
- Reprogram: with pads and pogo pins
- Data Trans.: Bluetooth or Zigbee  
SD card
- GUI: user interface to receive data

## First Prototype:

- soldering board
- with plug & cable
- with plug & cable
- with plug & cable

# Plan for Future Milestones

- Week 7
  - Test 1st prototype underwater
  - Design PCBs for fabrication
- Week 8
  - Add more peripherals (SD card, Bluetooth, etc.)
  - Set protocol for data transmission
  - Develop GUI for downloading data
- Week 9
  - Assemble and pot the device
  - Test 2nd underwater

# Acknowledgment

- Thanks for the support from Prof. Ryan Kastner
- Thanks for the support from Dr. Brian Zgliczynski
- Thanks for the help from Eric and Nathan

# Q&A

**Thanks!**