POTD - Problem-based Alarm

Group 16:
Sherry Wu
Shirley Xu
Charlene Zheng

Getting up on time has always been a problem

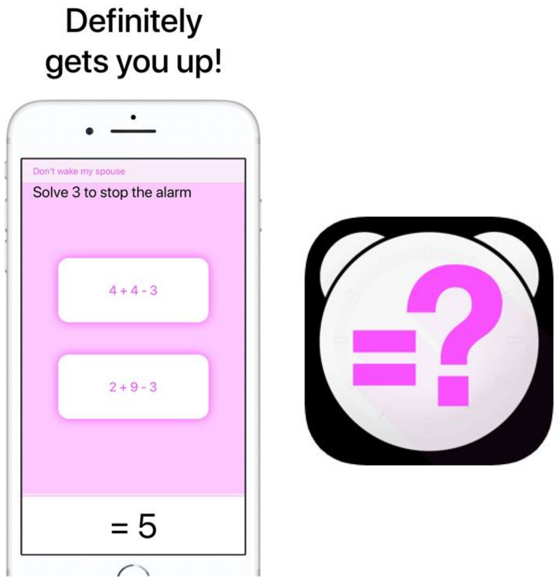
Problems

Turned off too easily

Battery dies in the middle of the night

Known solutions

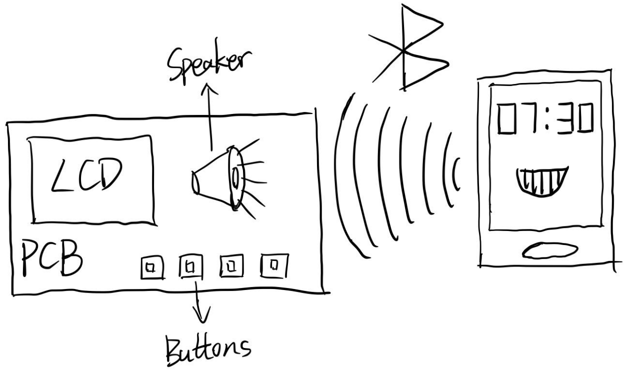
Would “run away” from people



Mathe Alarm

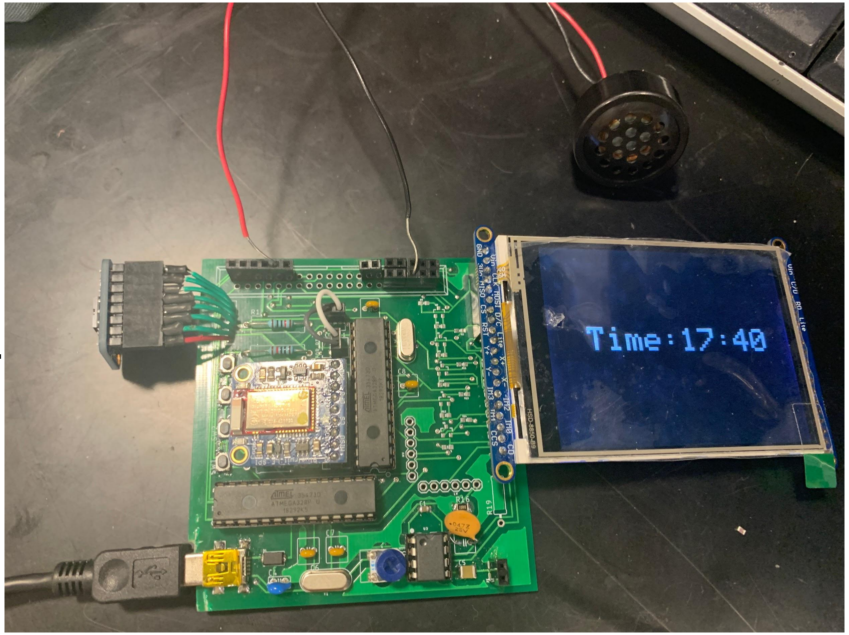
Our Solution: Problem-based alarm clock

- Answer multiple choice questions to turn off
- Customizable problems via Android app
- Feedback available



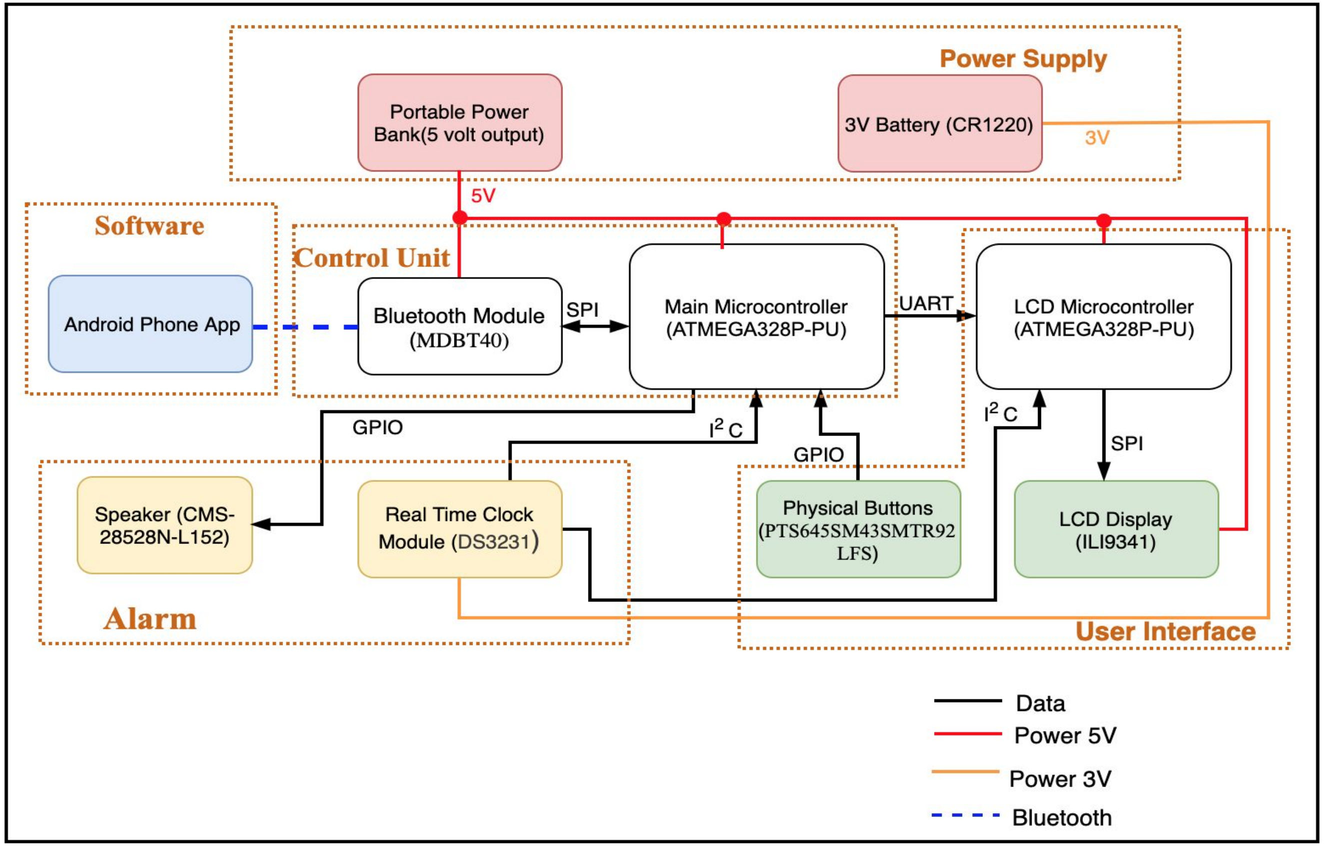
High level objectives

- Functions as an actual alarm clock
  - Display time
  - Beep at alarm time
- Displays questions and interacts with users.
- Communicates with Android app.
  - Set alarm time and questions
  - Review wake up time and "quiz performance"



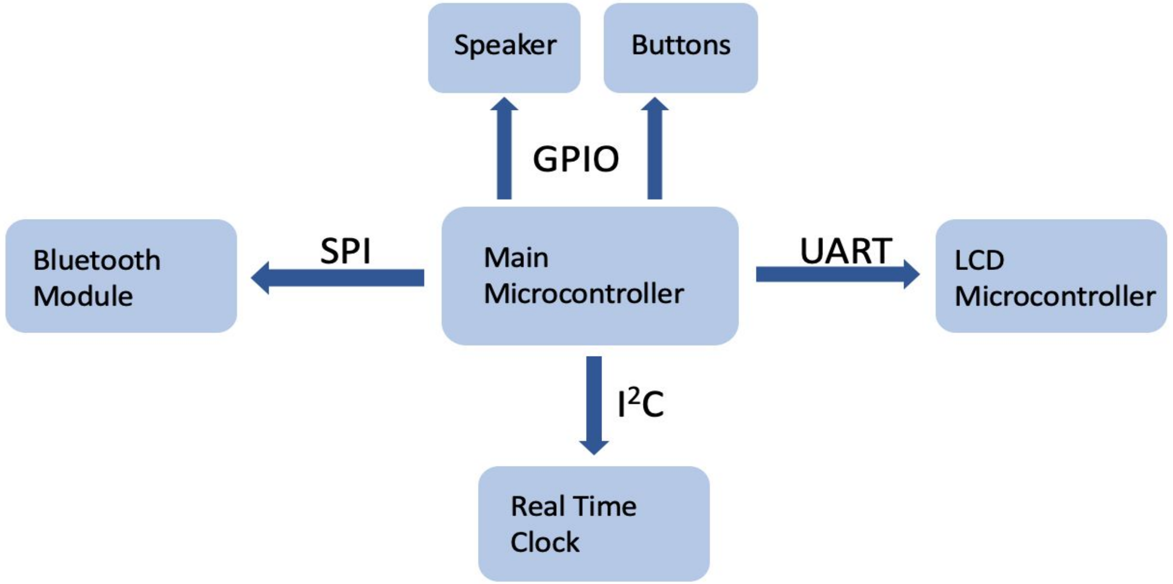
How it works

Block Diagram



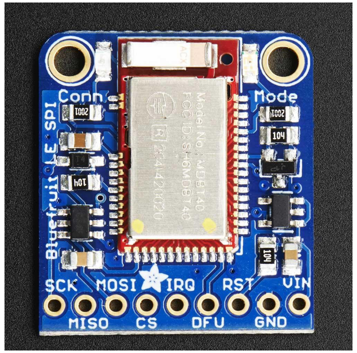
Power Supply
Portable Power Bank(5 volt output)
3V Battery (CR1220)
3V
5V
Software
Android Phone App
Control Unit
Bluetooth Module (MDBT40)
SPI
Main Microcontroller (ATMEGA328P-PU)
UART
LCD Microcontroller (ATMEGA328P-PU)
GPIO
I2 C
GPIO
I2 C
SPI
Speaker (CMS-28528N-L152)
Real Time Clock Module (DS3231)
Physical Buttons (PTS645SM43SMTR92 LFS)
LCD Display (ILI9341)
Alarm
User Interface
Data
Power 5V
Power 3V
Bluetooth

Control Unit - Main Microcontroller



Control Unit - Bluetooth

- Adafruit Bluefruit LE SPI Friend (MDBT40)
- Connect microcontroller and Android app
- Send data back and forth



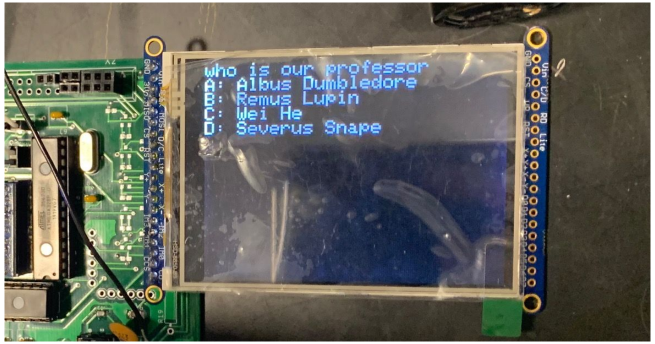
Power Supply

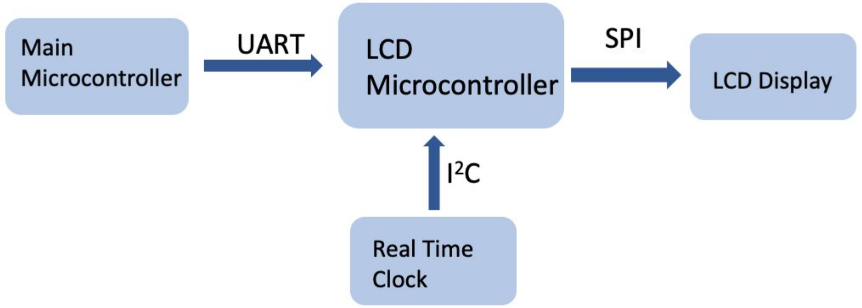
- 5V-10000mAh portable power bank
    - Original Design: two 9v Alkaline Batteries
        - Backlight current for LCD: ~ 80mAh
        - Duration :
            - 1160mAh/ 80mAh ~ about 14.5 hours
            - 10000mAh/80mAh ~ about 5.3 days
    - Connected via mini USB port

- Button battery for Real Time Clock
    - Continuous timekeeping when alarm is off

User Interface - LCD Microcontroller

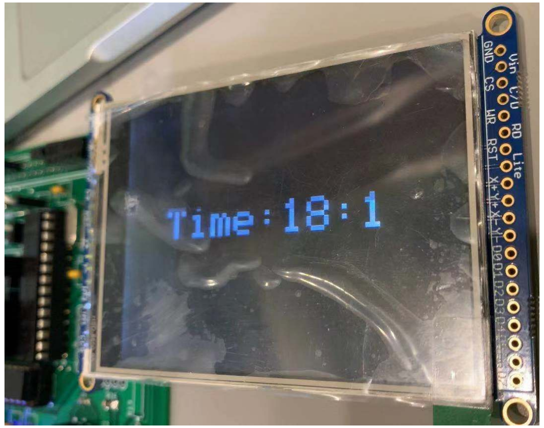
- Atmega328p as LCD controller
- Receive questions from the Main microcontroller via UART
- Constantly read time from RTC and display
- Send questions to LCD at alarm time





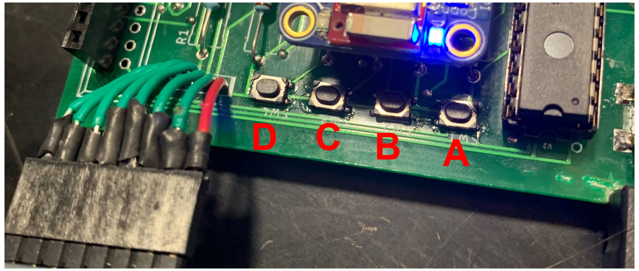
User Interface - LCD Display

- Used ILI9341 LCD display
- Receive data from LCD controller via SPI
- Display data on screen



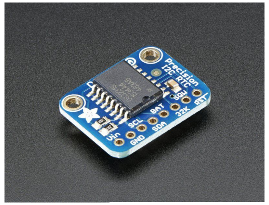
User Interface - Button

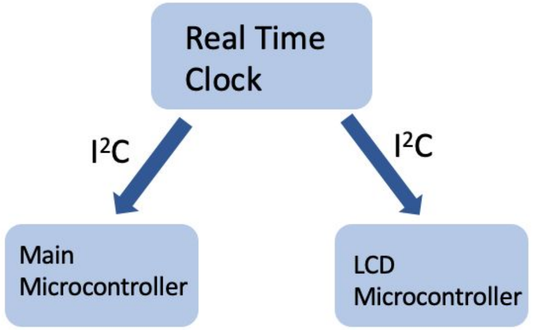
- Four buttons(PTS810 SJM 250 SMTR LFS)
- Send signal to Main microcontroller when pressed



Alarm - RTC (Real TIme Clock)

- DS3231
- Tracks time
  - Backup battery for continuous timekeeping
- Send time data to both microcontrollers via I²C.
  - Pull-up resistors





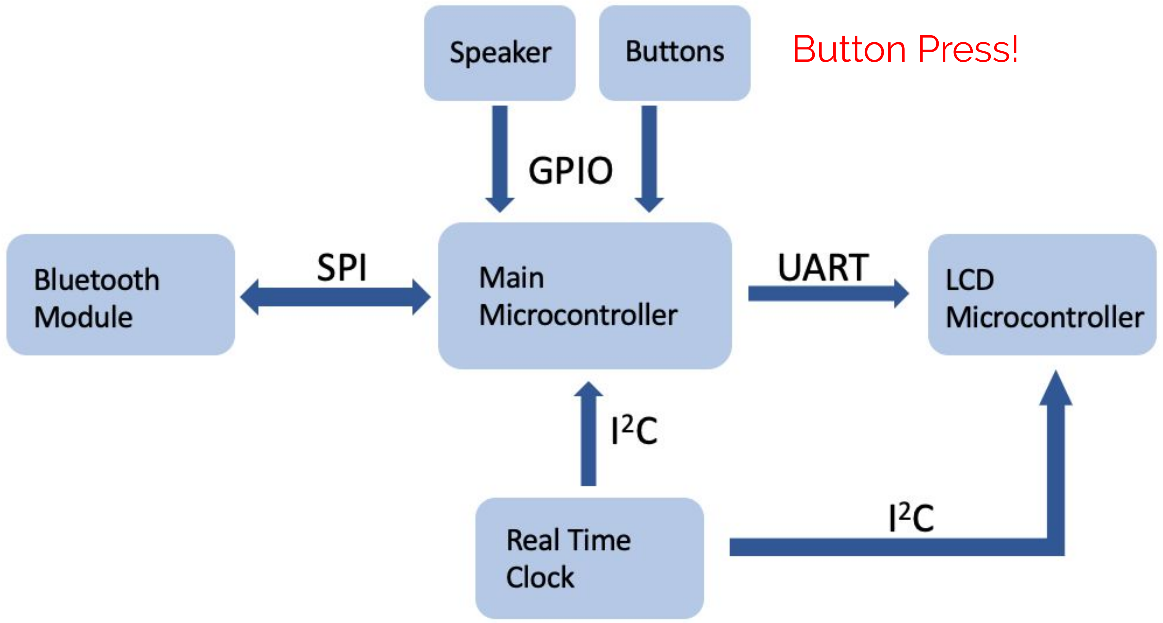
Alarm - Speaker

- Connected to Main Microcontroller via GPIO

System Integration Challenges

Button Press!

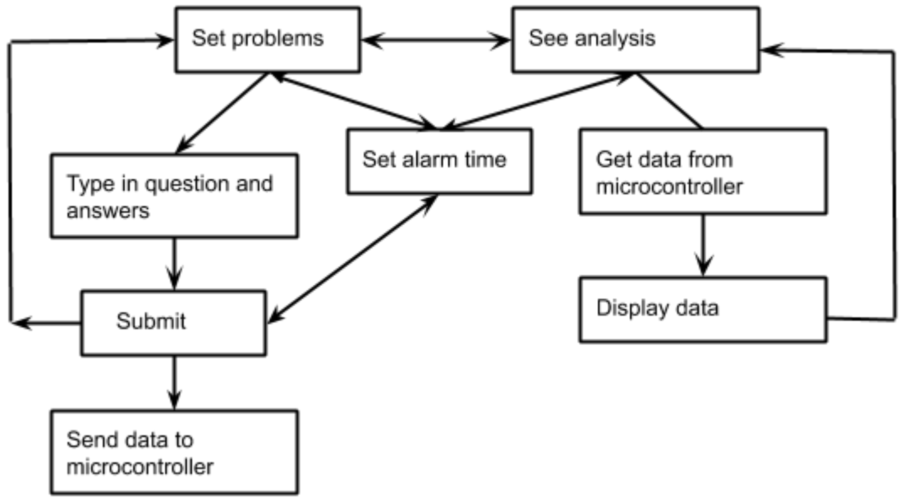
Data Loss!



Data Loss!

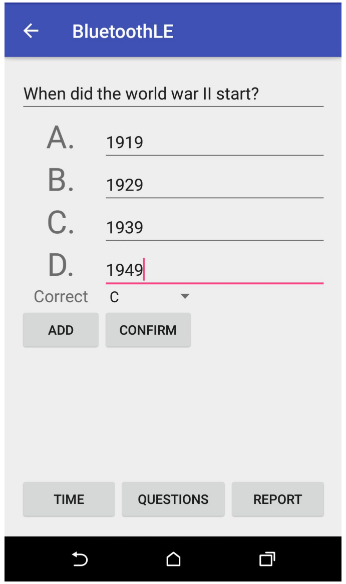
Android Mobile Application

- Connects to Bluetooth module
- Send and receive data to and from the microcontroller
- Found:
    - BLE could not send > 20 Bytes
    - Resolved by sending substrings


Set problems
See analysis
Type in question and answers
Set alarm time
Get data from microcontroller
Submit
Display data
Send data to microcontroller

Android Mobile Application

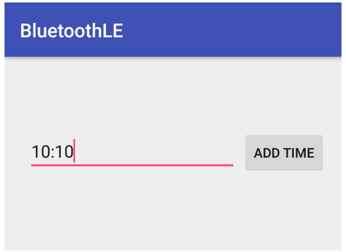
- Set alarm time
- Input questions
- Review data



BluetoothLE

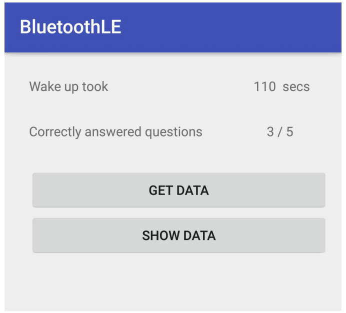
When did the world war II start?

A.	1919
B.	1929
C.	1939
D.	1949
Correct	C



BluetoothLE

10:10



BluetoothLE

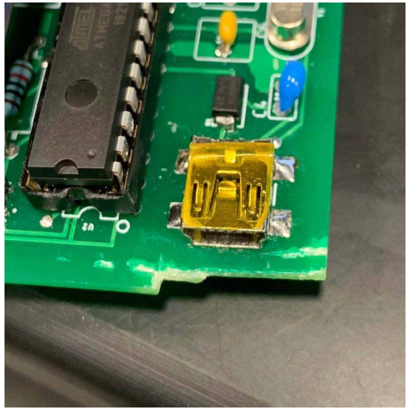
Wake up took	110 secs
Correctly answered questions	3 / 5

Conclusion

● Integrated all subsystems correctly & on schedule	
● PCB & system design	~2 weeks
● Having all subsystems work (on breadboard)	~2 weeks
● System integration on PCB	~1 week
● Implementing and testing the Android app	~1 week

Further Work

- Re-design PCB (position of mini usb)
- Implement IOS version of app
- Different alarm ringtones
- Multiple alarm times
- Better physical wrapping



Questions?