**Electronics Assignment: Portable Multi-Sensor Embedded System**

By – Abhishek Kumar

(Everything in this assignment is as per my practical work experience in electronics and embedded domain).

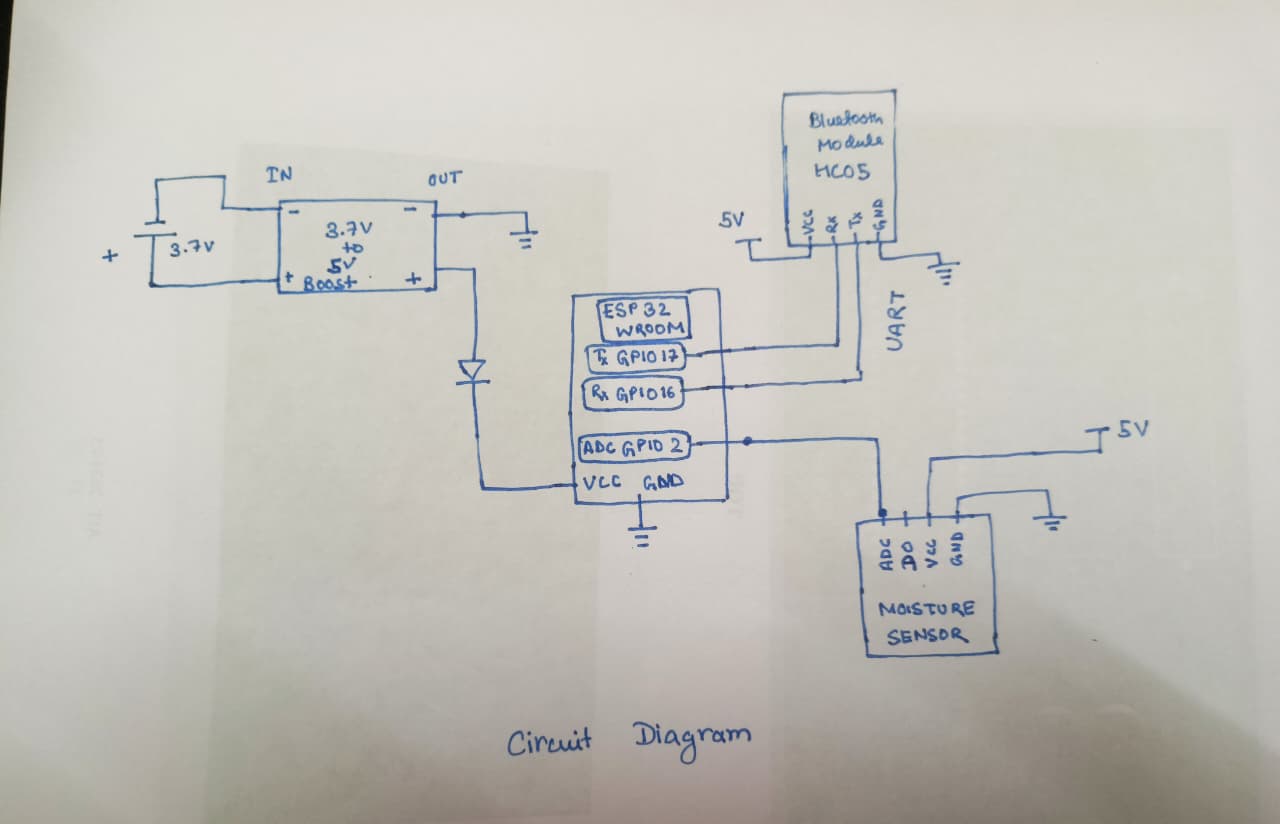
So here is the solution to the asked the solution, it is based on my everytime approach taken care while designing any embedded system.

I generalize any embedded system as, “**any input processed for output**”: here the input will be moisture level from a ‘moisture sensor’, output is logging data (no physical actuation). Input can be Digital High/low or ADC value, but as per system requirement it will have to be ADC value.

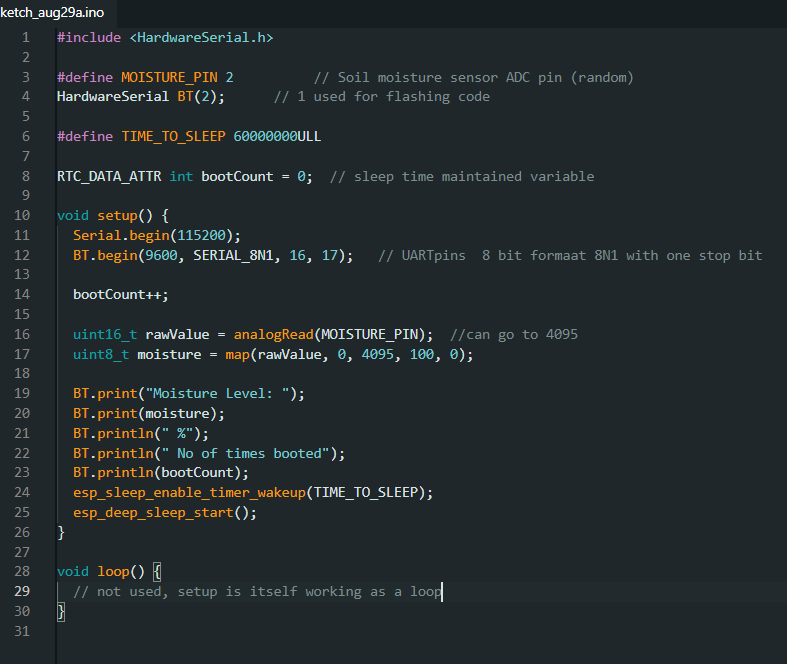
An additional thing is Debugging/viewing data- **UART based Bluetooth module is being used here.**

**CORE CIRCUIT DESIGN**

**Circuit Diagram**

****

**Basic CODE with ESP board installed on Arduino IDE, including HardwareSerial library for bluetooth module.**

****

**TO read data , pair your pc via Bluetooth to HC05 and open putty terminal. Check for COM port created by Bluetooth in Device manager. Enter the COM Port and baud rate as 9600 to see the output listed in code.**

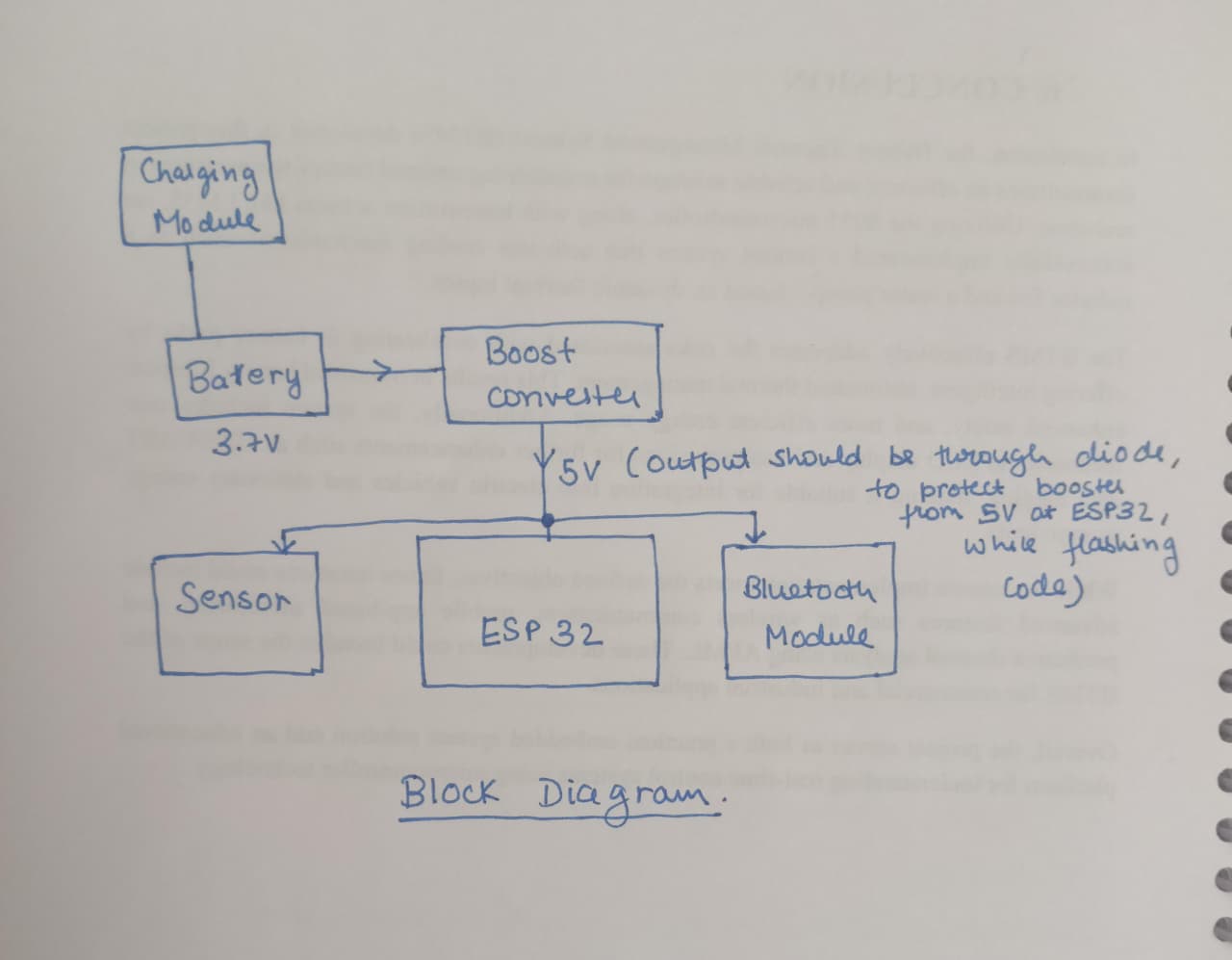
**Power:**

- 3.7v 2500mAh battery. ( which means will survive for 1hr if 2.5 A is drawn)

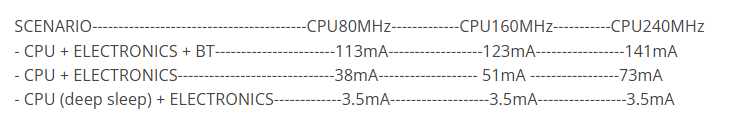
-A genweric 3.7v to 5v boost module will be used to power up Esp32 wroom, most sensor works on 5v/3.3v. Considering 5v to sensor also.

- As per assignment , esp32 will go in software initiated sleep to save power periodically.

**POWER CONSUMPTION**

****

**ESP POWER CONSUMPTION**

****

**Source -** <https://esp32.com/viewtopic.php?t=2662>

* Considering 240MHz operation, which is the default one.
* We are using CPU + Electronics mode, so 73mA or 0.073A will be used during active part and 3.5mA or 0.0035A.

HC05 – consumption varies 30mA – 70mA (idle,connected, data transfer)

Moisture sensor ~ 5mA.

1. HC05 and Moisture sensor is a generic component, so taking an average of HC05 drawn current as 50mA and moisture sensor current as 5mA. Total – 55mA
2. And Esp32 – 73mA (working mode).

Total Current – 128mA @ 5v @no sleep mode

Total current - 58.5mA @ 5v @ sleep mode

* **Current at 3.7v ( 3.7v-4.2v Lipo voltage range, taking 4v)**

5v \* 0.128A = 4.0v \* X \* 0.9 ( booster efficiency)

X = 0.18 A @ no sleep mode

5v \* 0.0585 = 4v \* X \* 0.9

X = 0.08 A @sleep mode

**BACKUP**

-As per battery consideration, it was 2500mAh

Means If 2.5A drawn, backup will be 1 hr

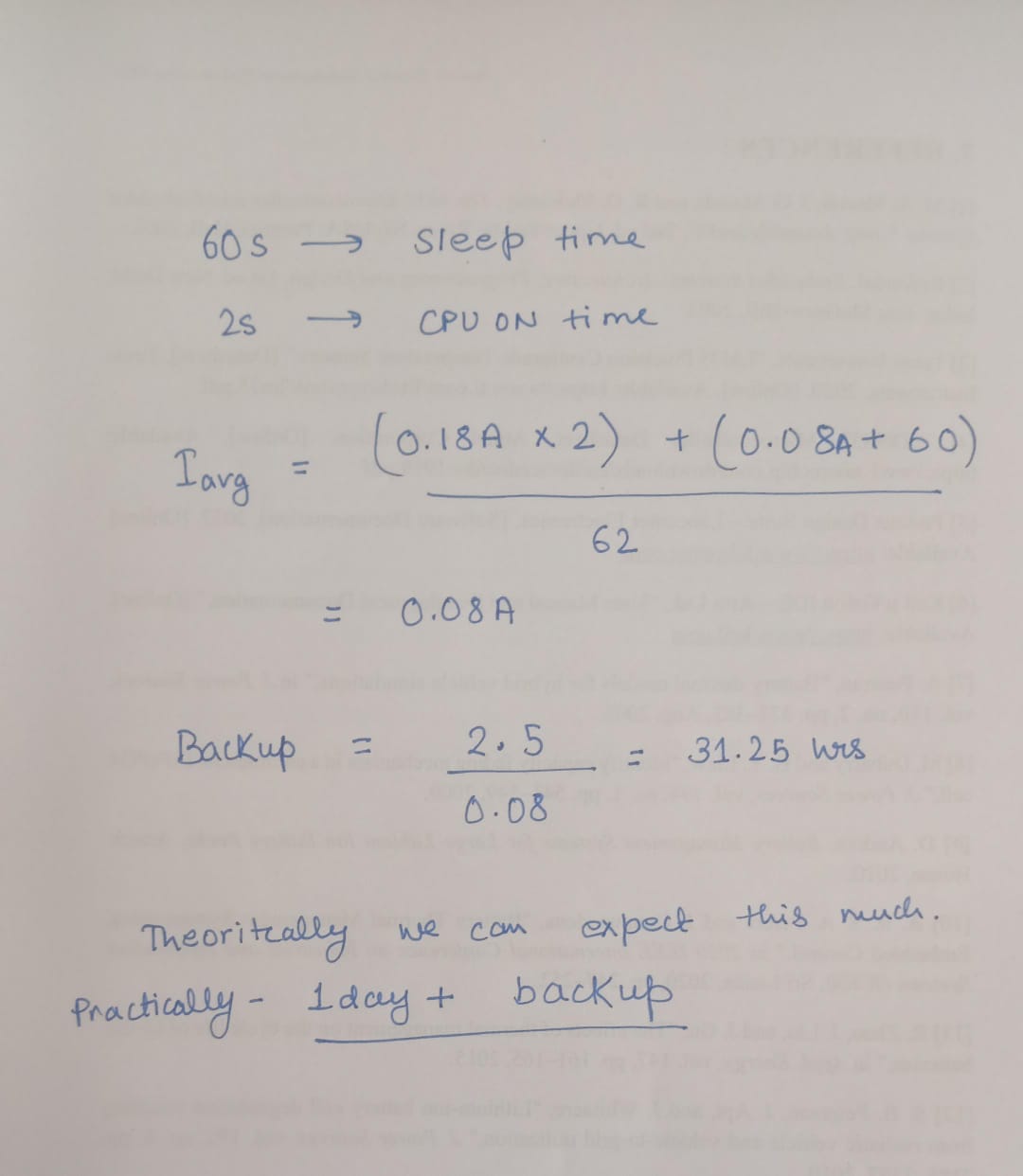
For 1 A 2.5hr

If 0.18A drawn, backup will be ( 2.5/0.18) = 13.888 hr

**13.89 hr will be backup will be without sleep mode. ( Duty Cycle 100%)**

**Here the Duty Cycle is 100%, now we can adjust Duty Cycle (Sleep Mode + Active Mode), to get desired backup time.**

**SAMPLE CALCULATION -**

****

**-This is for 2.5 Ah Cell, CPU operating at 240MHz**

**Power Optimization**

* The major power will be saved through hardwares, Remove unnecessary LED ( or just blink it at start, or any interval to indicate work.
* A good boost converter to reduce losses while converting 3.7v to 5v.
* Can use a driver circuit to HC05, as during ESP sleep mode it is still ON, it should be off. But this is not a feasible idea because reconnection every time, and boot time of HC05 .It can be considered for sensor, but again it must checked practically.
* A better practice will be to replace HC05 external Bluetooth module to ESP BLE.
* 240 MHz can be lowered to 80MHz, but in future if any calculation or more processing Job will be given it to ESP, it might cause problem.
* WIFI is not required, so power is saved since wifi is never initialized.
* Shortest possible active window, currently we are considering 2s, but it can be reduced further, it must be checked practically. ( in code 2s is not used it immediately enters sleep mode)

**Scalibility Note –**

So any intial prototyping should be done in versions,

**Prototype I** – should be based on basic modules( ESP32, Bluetooth, Moisture Sensor) enclosed in a basic Plastic box.

* This will give an idea of all the integration problems, that should be handled. Also give an idea of correct data coming at every edge cases Bit corruption/data mismatch, during communication like UART I2C, also gives idea of memory management.

**Prototye II** - A robust version with a common pcb, more SMD components this time, more robust case with some IP ratings, A good branded sensor with proper datasheet and documentation.

This should continue as per the output product we want , or the timeline, or the budget.

If we want a final robust version, we should be aware of all the component at base level, but we should get it done through good vendors in that particular field, like a good case will not be a 3D printed everytime, injection moulding case will definitely had its own benefits and robustness. ( Again this will be as per Budget limitation)

As per the timeline Of 6 months , I would say–

1st month – Design procurement of sensor and modules, making Prototype I, learning side by side.

2nd Month - Field testing of first and improving next prototype.

3rd-4th Month – Logging Final Proto data,with field testing, improving signal through software/hardware based filter, designing Test cases to pass the device.

5th – 6th Month – Final Product 100 unit production/Assembly.

Since units are 100, im considering manual testing of each product.

\*Still I have not been into production solely, but these are things which I think should be considered from R&D perspective.