**Remote Wireless Sensor Node Project**

Think about a scenario in which you want to gather some information constantly in the middle of a forest or a desert. One way you can do it is to be there yourself and try to collect the data twenty-four-seven, which could be very troublesome and inconvenient. Another way you can solve this is by using a remote wireless senor node. A wireless sensor network consists of nodes that are able to gather sensory information, process it, and communicate it through other connected devices in the same network. By using a wireless sensor node, you are able to stay at home and gather the data you need from wherever you want.

A very good example is a solar powered fire detection system that can detect forest fires. The entire transmitter system, powered by a solar panel, is located in the forest, and contains many different sensors, a microcontroller, and a transmitter. The receiver side at the fire station continuously monitors the signals and alerts the fire fighters whenever a fire is detected. We are going to build a prototype similar to that for this project.

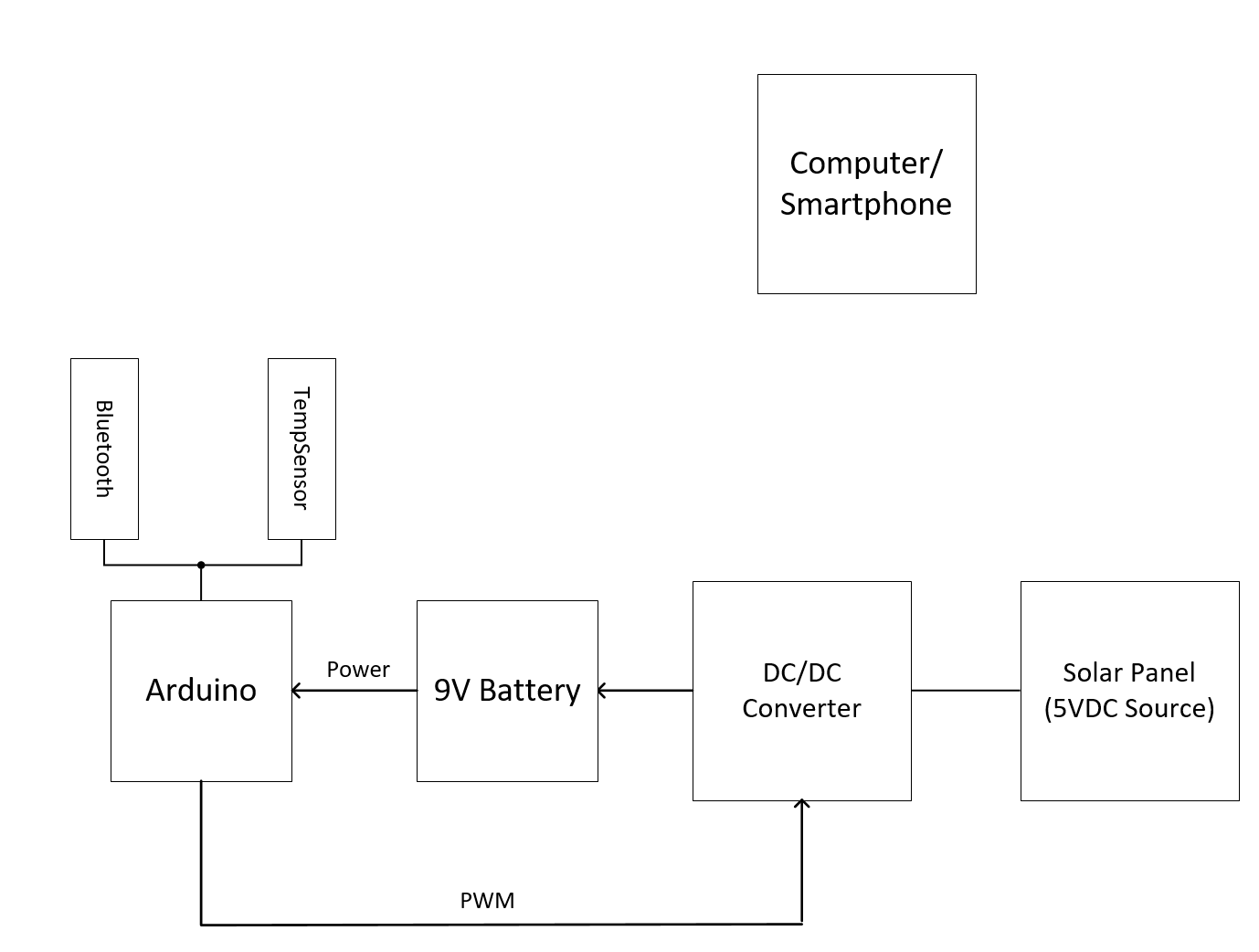
**Project Description**

In this project, you are going to build a remote wireless senor node. This project requires both hardware and software designs. The overall system is shown in the block diagrams below. On the transmitter side, we have a solar panel, a temperature sensor (thermistor), a Bluetooth module, a DC/DC converter that regulates the power from the solar panel to an Arduino. On the receiver side, we have a computer or your smart phone to display the temperature information.

In this project, we are going to use the 5V power supply from AD2 as our solar panel (input voltage could vary). The DC/DC converter circuit, however, needs to regulate the output at a constant voltage to power up the Arduino, through a pulse width modulation (PWM) feedback control with a switching frequency greater than **5 kHz**. In addition, conduct an optimal load test to find out the **efficiency** of your power stage. Your Arduino should also process the information from the temperature sensor and transmit the data through Bluetooth. In order to save power, if the battery is fully charged, your Arduino should process and send the temperature information every 30 seconds, and it goes to sleep for the rest of time; if the battery is not fully charged, your Arduino should not go to sleep. In order to determine whether the battery is fully charged, it is a good practice to run a **battery charging/discharging test**. Your computer or smart phone on the receiver side should display the temperature readings. Enjoy the project!



In this block diagram, the solar panel is going to charge a 9V battery through a DC/DC converter. The battery provides power to the Arduino, which in addition to processing temperature information will provide PWM to the DC/DC converter.



Note: if the rechargeable battery is not at 9V, please use your DC/DC converter to charge the battery first. The Arduino requires 9V to work. The only library you can use for this project is the one for Bluetooth communication.

Deliverables specific to this project:

1. A working prototype that satisfies the requirements discussed above.

2. A working DC/DC converter with PWM feedback control: demonstrate that under different input voltages, its output voltage is still regulated at desired voltage.

3. Run an optimal load test on your DC/DC converter and plot its efficiency curve. The y-axis of the efficiency is the DC/DC converter efficiency, and the x-axis is the load resistance.

4. Show battery charging and discharging waveforms. Explicitly showing the mAh in and out of the battery.

Enjoy the project!

Resources:

1. DC/DC Converter Steady State Analysis Book Chapter

2. MCP9700 temperature sensor data sheet:

<http://ww1.microchip.com/downloads/en/devicedoc/20001942g.pdf>