Arduino Model Rocket

# Objective

Create a small model rocket that:  
- Measures and logs flight data (altitude, acceleration)  
- Automatically deploys a parachute at the right time for safe landing  
- Uses Arduino to control and automate everything

# Components Needed

- Arduino Nano / Uno: Brain of the rocket, reads sensors and controls outputs

- BMP280 Sensor: Measures air pressure → calculates altitude

- MPU6050: Detects motion, acceleration, and orientation

- SD Card Module + SD Card: Saves flight data for later analysis

- Servo motor: Releases the parachute mechanically

- LiPo battery / Power source: Powers the system

- Model rocket kit: Houses electronics and propulsion system

- Solid propellant engine: Propels the rocket (solid fuel)

- Launch controller: Used to safely ignite the rocket motor

# How the System Works

## Pre-Launch

* - Upload code to Arduino
* - Power on the system
* - Arduino reads baseline altitude (ground level)
* - System arms itself and begins logging data

## During Launch

* - Rocket launches via solid propellant motor
* - BMP280 detects rising altitude
* - MPU6050 detects strong vertical acceleration

## At Apogee (peak)

* - Altitude stops increasing
* - Acceleration drops (free-fall begins)
* - Arduino detects falling altitude and negative acceleration
* - Arduino activates servo motor to release parachute

## Descent

* - Parachute slows rocket down
* - Sensors continue logging data
* - Rocket lands safely

# Wiring Overview

- A4 (SDA): BMP280 & MPU6050

- A5 (SCL): BMP280 & MPU6050

- D9: Servo Signal Pin

- D10: SD Card CS

- D11-D13: SD Card SPI Pins

- 5V / GND: Power to all modules

# Parachute Deployment Mechanism

A simple servo-based system:  
- A hatch on the rocket tube is held shut with a pin  
- The parachute is tied to a spring or elastic band to push it out  
- The servo horn pulls the pin when triggered  
- When apogee is detected, the Arduino moves the servo to release the parachute

# Arduino Code (Simplified)

#include <Wire.h>  
#include <Adafruit\_BMP280.h>  
#include <MPU6050.h>  
#include <Servo.h>  
#include <SD.h>  
#include <SPI.h>  
  
Adafruit\_BMP280 bmp;  
MPU6050 mpu;  
Servo chuteServo;  
  
const int chipSelect = 10;  
float groundAltitude = 0;  
bool chuteDeployed = false;  
  
void setup() {  
 Serial.begin(9600);  
 if (!bmp.begin()) { while (1); }  
 mpu.initialize();  
 if (!SD.begin(chipSelect)) { while (1); }  
 chuteServo.attach(9);  
 chuteServo.write(0);  
 delay(2000);  
 groundAltitude = bmp.readAltitude(1013.25);  
}  
  
void loop() {  
 float altitude = bmp.readAltitude(1013.25);  
 float accelZ = mpu.getAccelerationZ();  
 File dataFile = SD.open("flight.txt", FILE\_WRITE);  
 if (dataFile) {  
 dataFile.print("Altitude: "); dataFile.print(altitude);  
 dataFile.print(", AccelZ: "); dataFile.println(accelZ);  
 dataFile.close();  
 }  
 if (!chuteDeployed && altitude > groundAltitude + 50 && accelZ < 0.5) {  
 chuteServo.write(90);  
 chuteDeployed = true;  
 }  
 delay(100);  
}

# Safety Tips

* - Always test electronics on the ground first
* - Do NOT use explosive charges unless trained
* - Use low-power motors for test flights
* - Never fly near people, roads, or animals
* - Use bright-colored parachutes to help find your rocket

# Future Add-ons

* - GPS tracker module (NEO-6M) for real-time location
* - LoRa or RF module to transmit data to ground station
* - OLED display to show data post-flight
* - A camera (like ESP32-CAM or 808 keychain cam)