Smart Cart
An Enhanced Shopping Experience

Team 41
Kartik Sanghi
Nikhil Raman
Rohan Singh

Introduction

- The Smart Cart provides an efficient shopping checkout system cutting down waiting times at the counters
- The system combines commonly available hardware to provide a cost effective solution for our sponsors Wal-Mart

Features

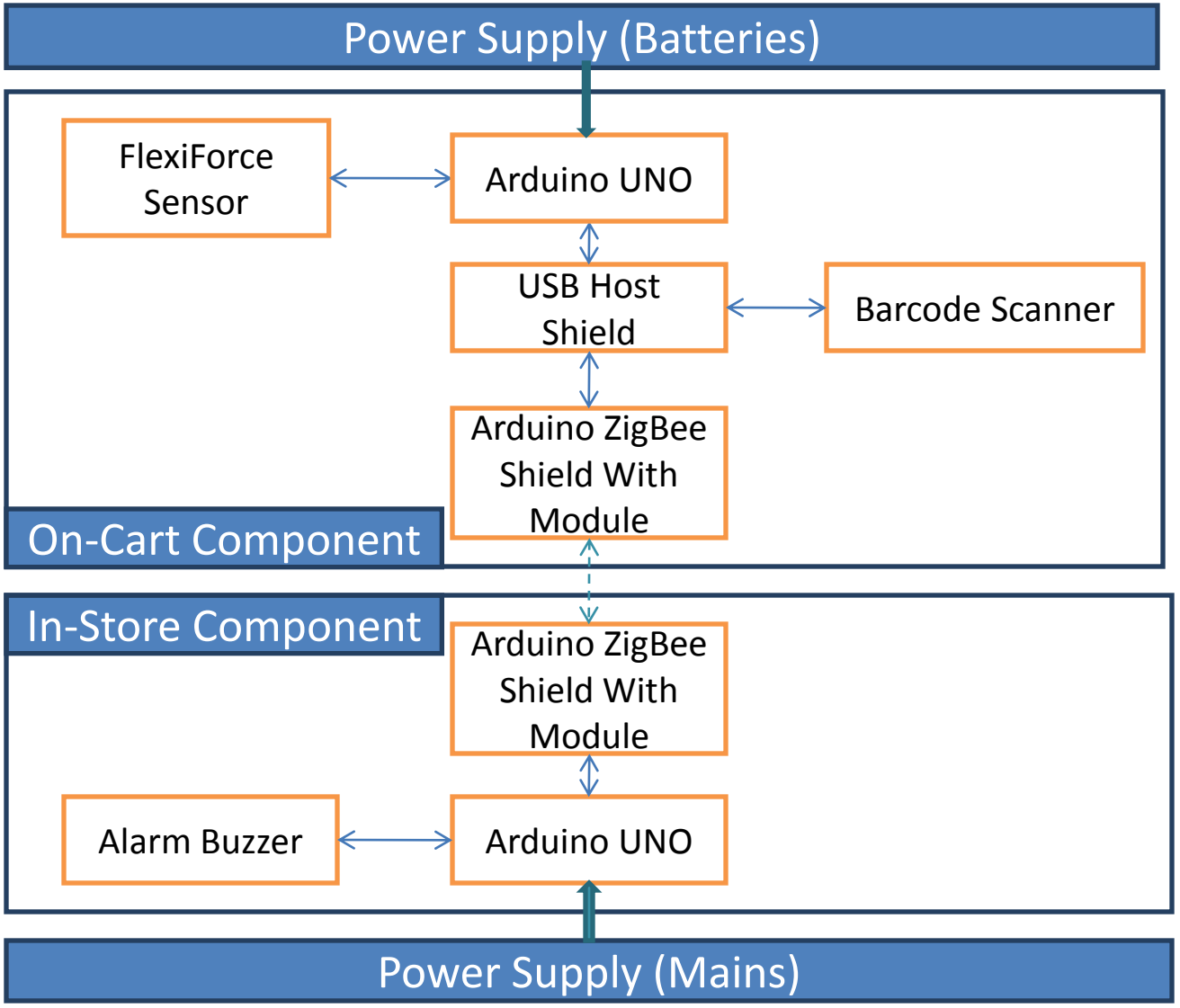
- Compatibility with various barcode systems
- Real time weight monitoring to detect pre-payment theft
- Robust communication system between cart and store to detect non-payment theft
- PC/LCD Serial Interface
- Modular and expandable design

The Smart Cart

System Overview

- Hardware
  - Alarm buzzer circuitry
  - PC/LCD Interface
  - FlexiForce weight sensor circuitry
  - ZigBee wireless modules
- Software
  - Arduino control for barcode reading and matching
  - Arduino program for weight monitoring
  - Alarm triggers

Block Diagram



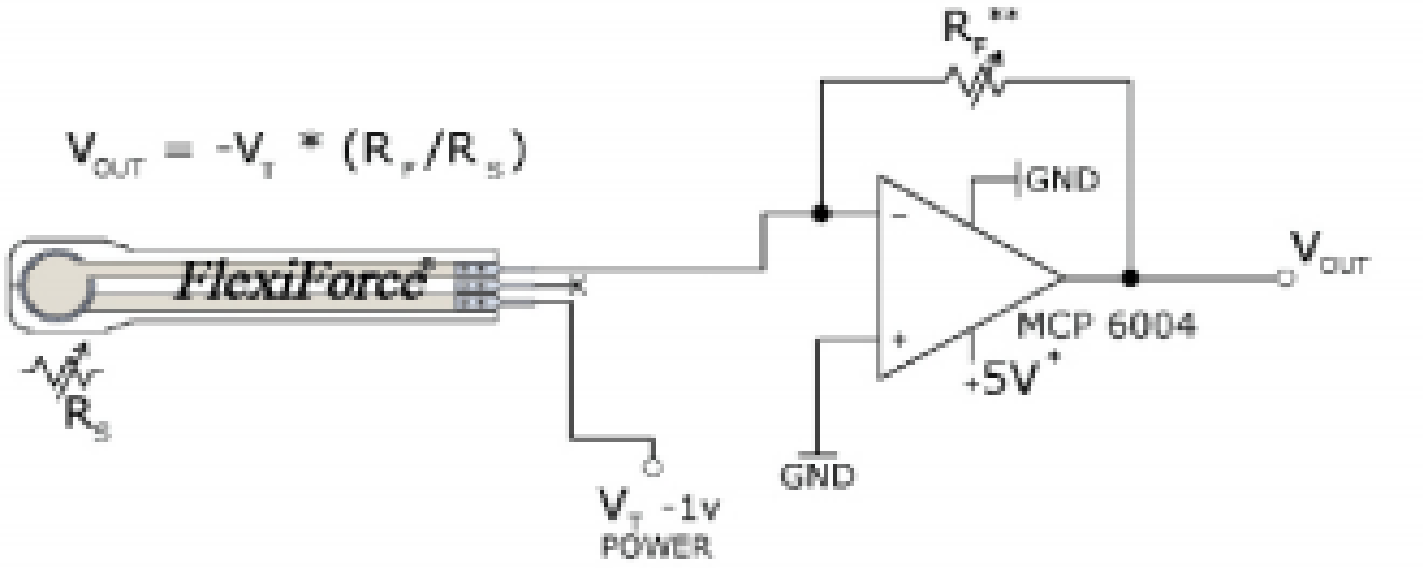
Hardware Overview

- FlexiForce circuitry
  - Takes non linear sensor output and converts it to a linear input at the Arduino
- Alarm buzzer
  - Sets of an alarm upon receiving a defined signal from the Arduino
- ZigBee modules
  - Take certain data from the Arduino microcontroller serial output and communicate the data wirelessly

FlexiForce Sensor Circuitry

- FlexiForce Sensor
- LM324N Single Rail Op-Amp
- Sends output to Arduino

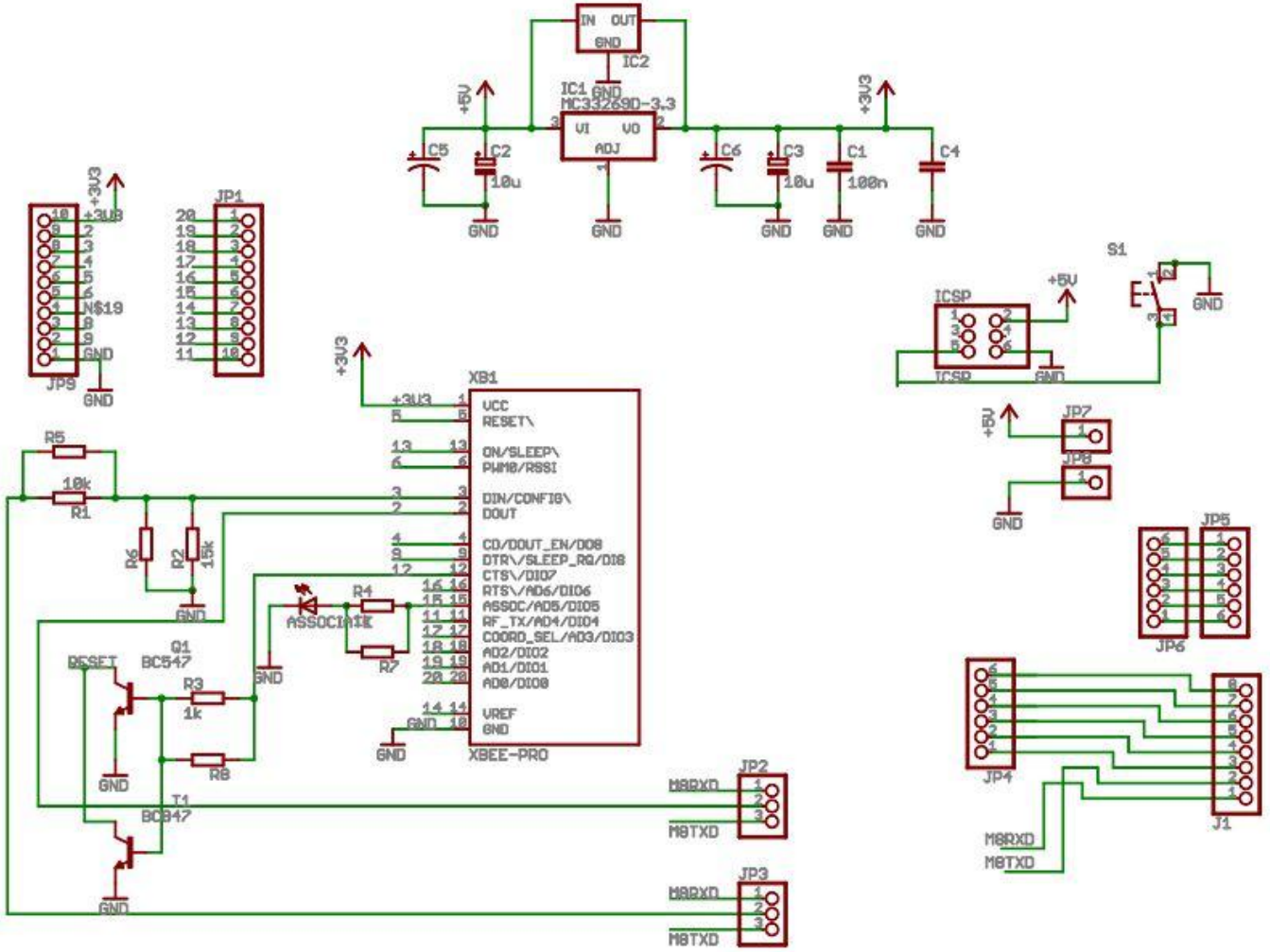
FlexiForce Sensor Circuit Schematic



ZigBee (Xbee) Wireless Modules

- Takes serial output from Arduino
- Establishes serial wireless communication with second module
- Sends output to Arduino

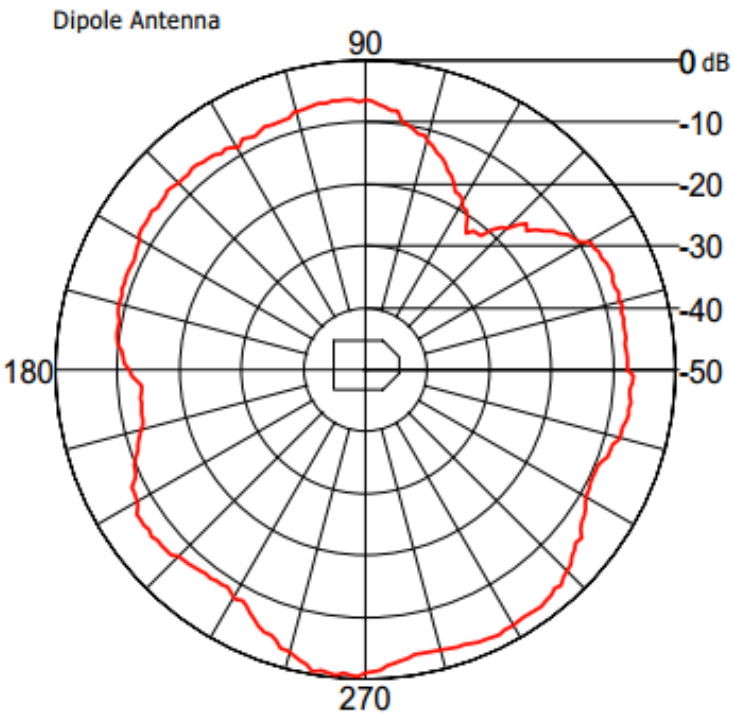
ZigBee Module Schematic

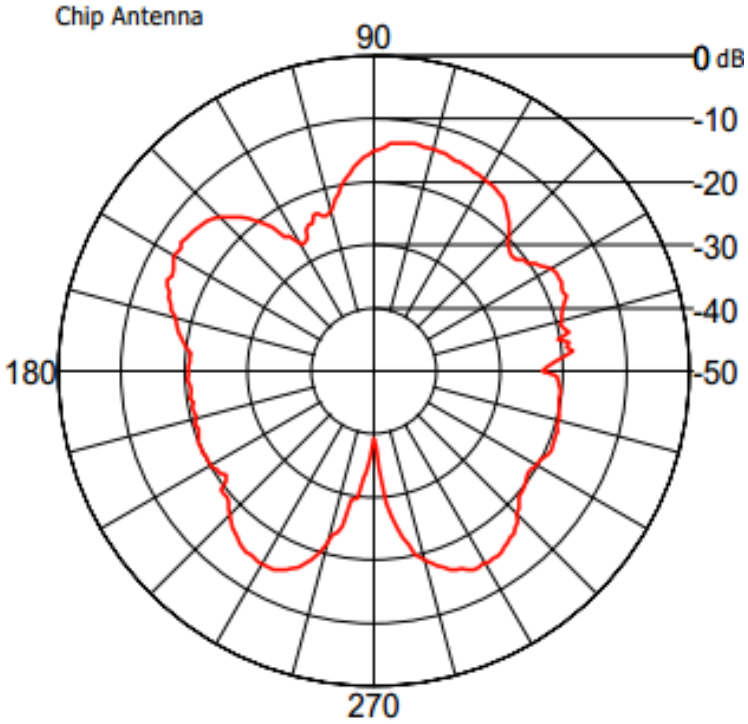


Factors Affecting ZigBee Communication

- Background noise
- Other ZigBee networks
- Wi-Fi and Bluetooth

Antenna Comparison (Xbee)

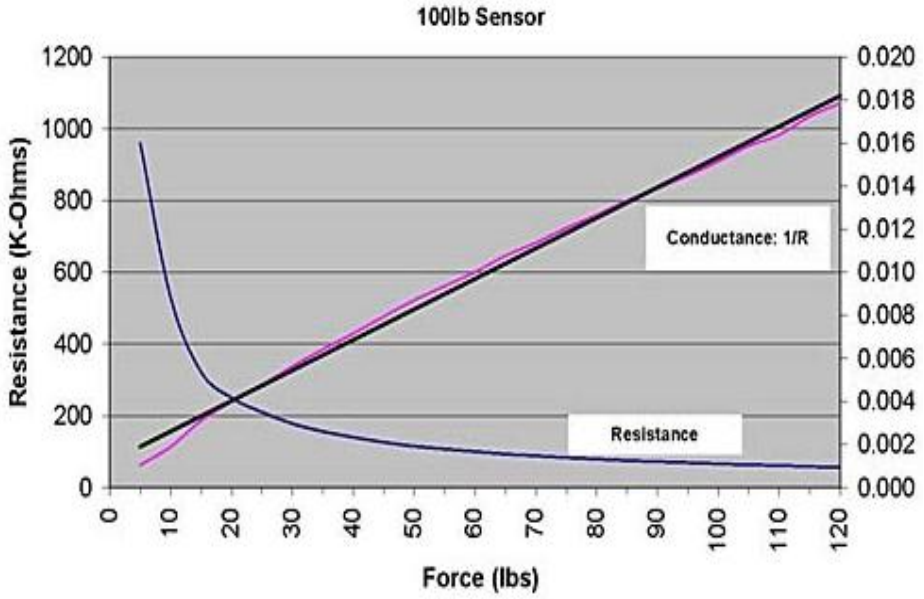


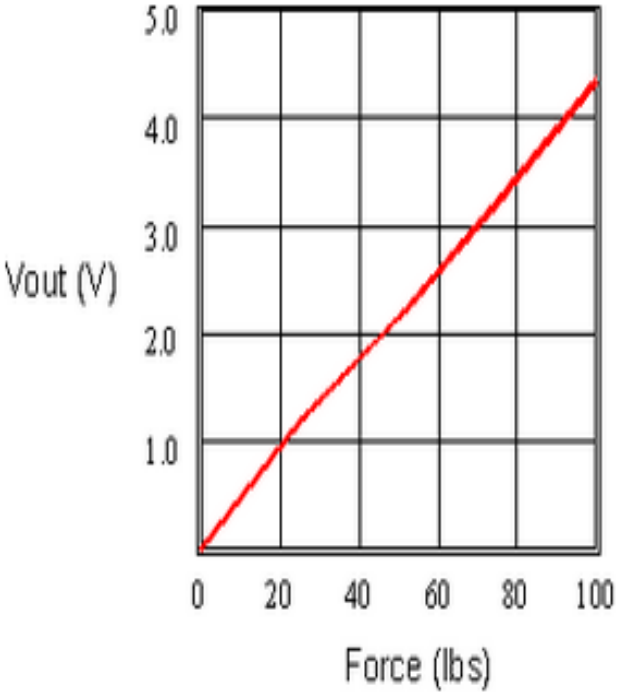


Xbee Testing Results (Error)

Xbee Testing Results (Error-Free)

FlexiForce Sensor Operation



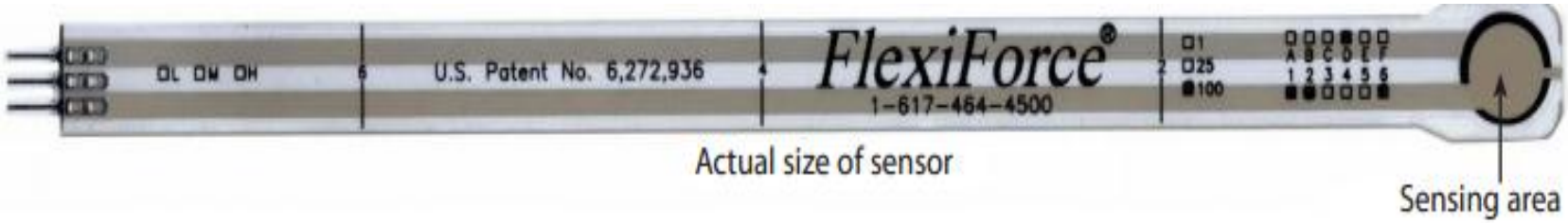


Factors Affecting Sensor Sensitivity

- Conditioning
- Feedback resistor
- Object placement

Sensor Recommendations

- Condition sensor with 110% of rated weight
- Place sensor on flat surface
- Object must be on sensitive area
- Place object on a puck to concentrate weight



Why Condition the Sensor?

- Linearity errors
- Greater hysteresis
- Greater response time

TYPICAL PERFORMANCE	
Linearity Error	<±3%
Repeatability	<±2.5% of full scale
Hysteresis	<4.5% of full scale
Drift	<5% per logarithmic time scale
Response Time	<5 microseconds

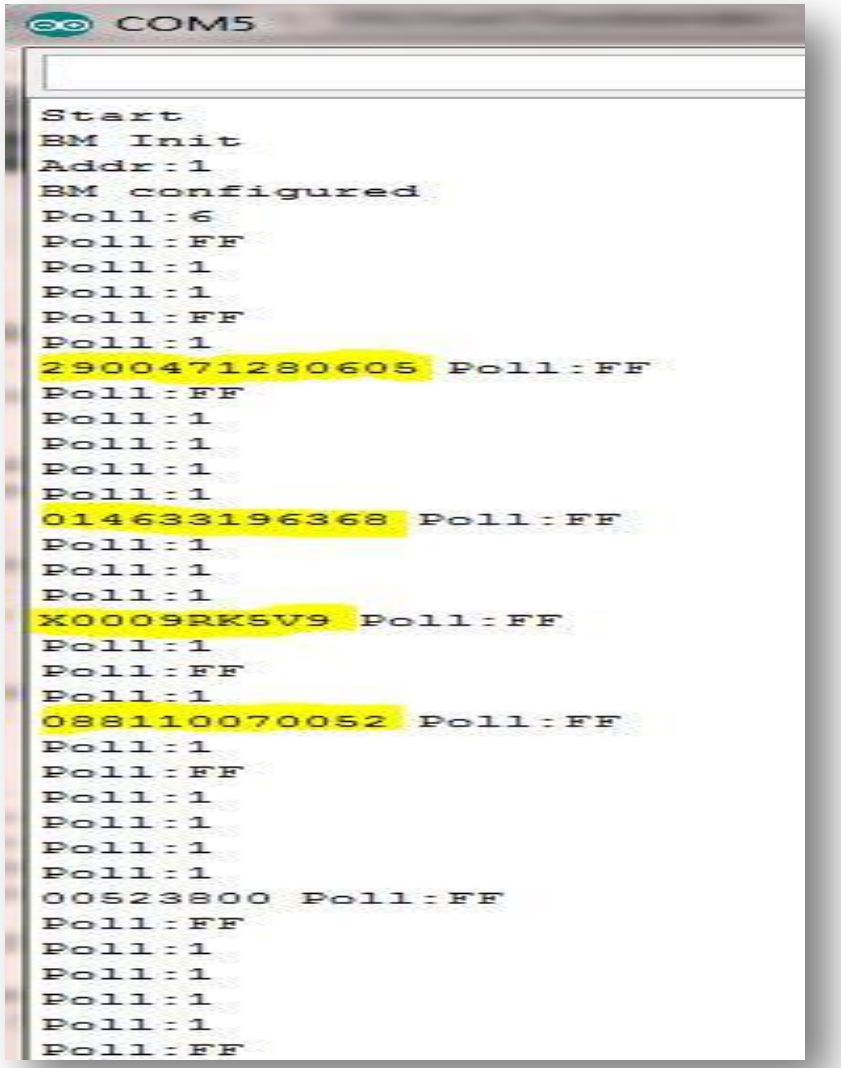
Software Description

- Arduino Software
  - Barcode read
  - Product Info Retrieval
  - Weight monitoring
  - Xbee module control
  - Alarm triggers

Barcode Read

- Scanner connected to Arduino through USB host Shield
- USB host shield uses SPI interface which requires digital I/O pins 8, 10 , 11 and 12 of Arduino
- Arduino receives data from the scanner through a UART serial communication port
- The UART transmits the bits in a sequential fashion

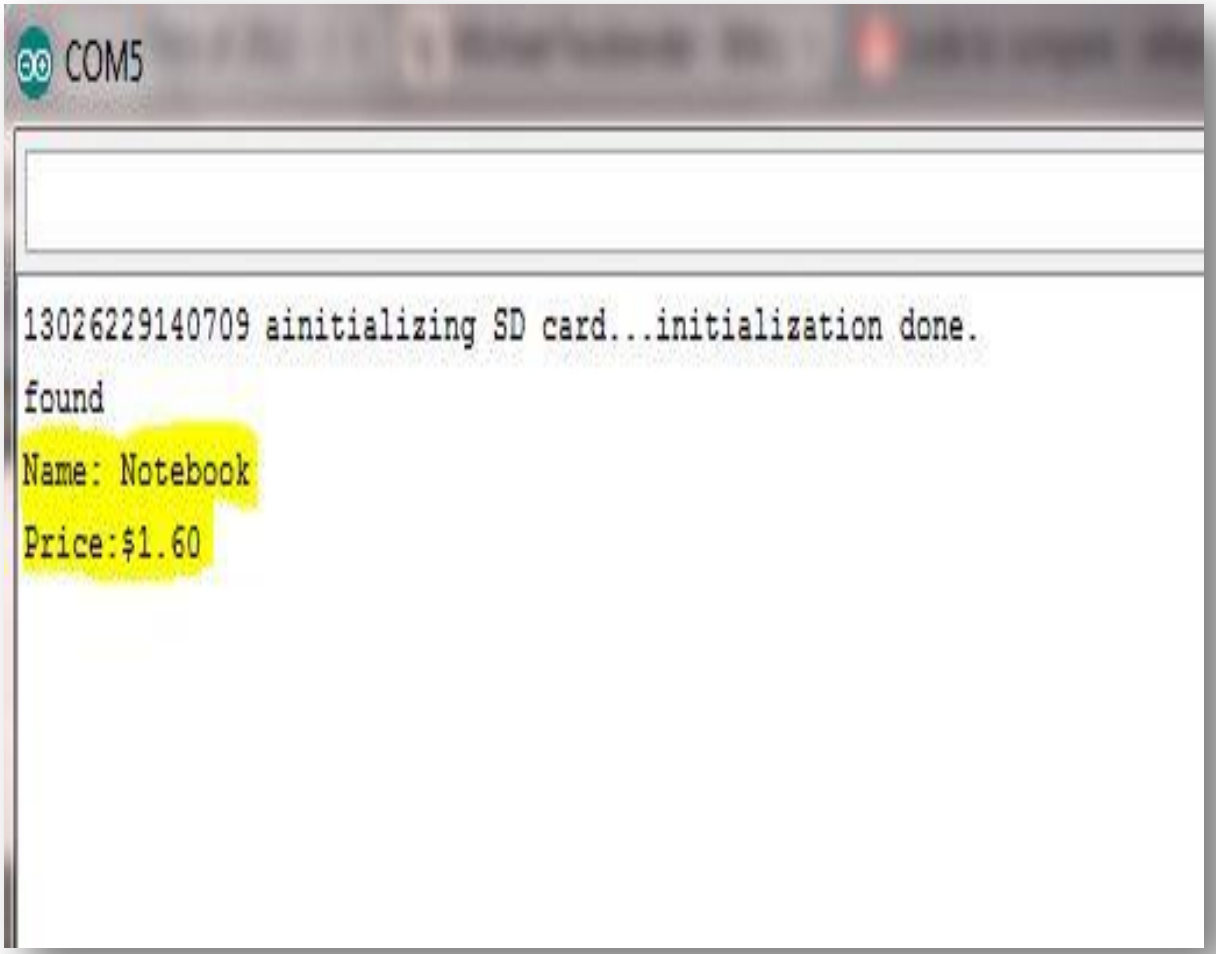
Barcode Read Output



Product Info Retrieval

- Use SD card to store product database
- Store current barcode scanned on Arduino EEPROM
- Compare the scanned barcode with barcodes stored in the database
- If compare successful, display product's attributes

Product Retrieval Output



Weight Monitoring

- Theft detection Criteria :

  Weight on cart > Total Scanned Weight
- If unscanned product is placed on the cart, then the sensors detect a weight mismatch which sets of the buzzer
- The buzzer doesn't stop until the weight discrepancy is fixed which requires scanning the product or removing it from the cart

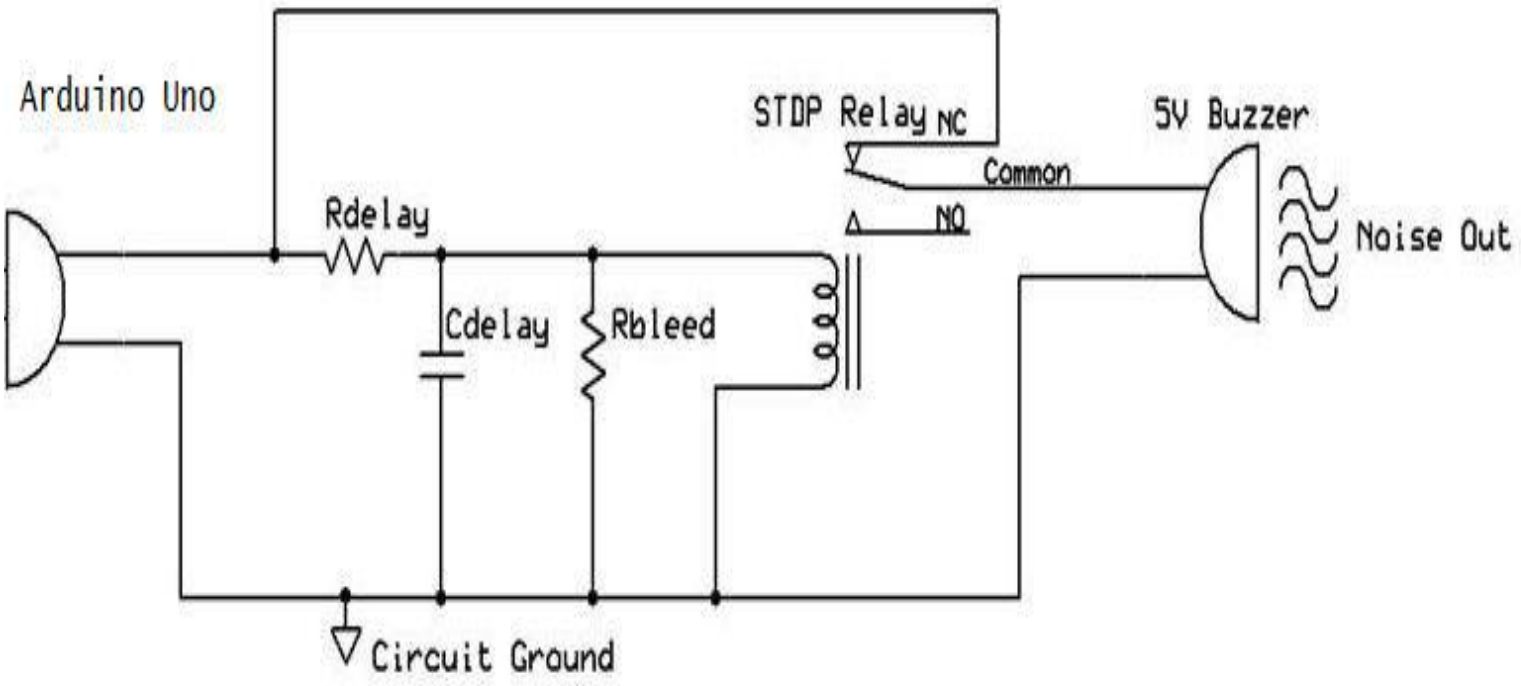
Alarm Trigger

```
void setup() {
  pinMode(4, OUTPUT); // set a pin for buzzer output
}

void loop() {
  buzz(4, 2500, 500); // buzz the buzzer on pin 4 at 2500Hz for 1000 milliseconds
  delay(1000); // wait a bit between buzzes
}

void buzz(int targetPin, long frequency, long length) {
  long delayValue = 1000000/frequency/2; // calculate the delay value between transitions

  long numCycles = frequency * length/ 1000; // calculate the number of cycles for proper timing

 for (long i=0; i < numCycles; i++){ // for the calculated length of time...
    digitalWrite(targetPin,HIGH); // write the buzzer pin high to push out the diaphram
    delayMicroseconds(delayValue); // wait for the calculated delay value
    digitalWrite(targetPin,LOW); // write the buzzer pin low to pull back the diaphram
    delayMicroseconds(delayValue); // wait againf or the calculated delay value
  }
}
```

Alarm Buzzer Circuit

- Takes an input from Arduino
- Buzzer sets off upon input
- Noise determined by Resistor connected to the buzzer
- Higher resistance gives lesser noise but low resistance can damage buzzer due to high current

Alarm Buzzer Schematic



Future Hardware Developments

- Improved battery life
  - Lower power barcode scanner
  - Set up Xbee on lower power rating
- Alternative sensor
  - Find a sensor with fewer sensing restrictions
  - Better high frequency characteristics

Future Software Developments

- Improved User Interface
  - Responsive to user input
  - More than just a display
- On cart payment Processing
  - Possible smartphone application
  - If possible, a hardware solution

Challenges Faced

- Limited scope of Arduino
  - Lack of interfaces with various protocols
  - Integration made difficult
- FlexiForce sensor
  - Limited sensing area
  - Non-linear output

SWOT Analysis

- Strengths
  - Modular design, expandable
  - Cost effective
  - Real time
- Weaknesses
  - Difficult integration
- Threats
  - RFID tag systems
  - Smartphone applications
- Opportunities
  - Fast lane for checkouts
  - Smart Cart

Credits

- Professor P. Scott Carney
- Justine Fortier, our TA
- Google

Thank you

- Questions???