Testing the Arduino Nano

The Arduino Nano is a rugged device, but when attached to an AEV, with motors, batteries and sensors, quite a bit can potentially go wrong. The Arduino Nano test board pictured below enables testing of the Nano in isolation, with none of the AEV parts attached to cause potential problems. The socket in the center is for the Nano, with the USB port indicated by an icon etched in the Plexiglas cover. (Cleverly concealed by an actual USB cable.) The row of blue and green LEDs is used to test the Digital I/O capability, blinking each one in turn to verify operation. The blue LEDs correspond to the PWM capable DIO lines, and those are individually faded in and out to verify the PWM capability. The analog inputs are verified by a set of CdS photocells to provide a variable voltage input to each AI port. There are two reset buttons to verify the reset capability, and a second set of buttons that test the 3.3V and 5V power.
Testing the Arduino Nano

The Arduino Sketch code for the test is as follows:

```cpp
// Initialize variables & constants for the program
int i; // Loop counter
int j; // Another loop counter
const int blue[] = {3, 5, 6, 9, 10, 11}; // PWM LEDs
int CdS[8]; // Array for CdS analog values

void setup() { // All DIO lines set to outputs
  pinMode(0, OUTPUT);
  pinMode(1, OUTPUT);
  pinMode(2, OUTPUT);
  pinMode(3, OUTPUT);
  pinMode(4, OUTPUT);
  pinMode(5, OUTPUT);
  pinMode(6, OUTPUT);
  pinMode(7, OUTPUT);
  pinMode(8, OUTPUT);
  pinMode(9, OUTPUT);
  pinMode(10, OUTPUT);
  pinMode(11, OUTPUT);
  pinMode(12, OUTPUT);
  pinMode(13, OUTPUT);
}

void loop() {
  // Blink the DIO line LEDs sequentially
  for (i = 0; i < 14; i++){
    for (j = 0; j < 10; j++){
      digitalWrite(i, HIGH);
      delay(100);
      digitalWrite(i, LOW);
      delay(100);
    }
  }

  // Pulse the PWM lines to fade in and out the blue LEDs
  for (i =0; i < 6; i++){
    for (j = 0; j < 255; j++){ // Make brighter
      analogWrite(blue[i], j);
      delay(5);
    }
    for (j = 255; j > -1; j--){ // Make dimmer
      analogWrite(blue[i], j);
      delay(5);
    }
  }

  // Open a terminal window to see the raw analog values
  // Putty is suggested for the Terminal emulator program
  Serial.begin(9600);
  for (j = 0; j < 1000; j++){ // Do this for a while...
    for (i = 0; i < 8; i++){
      CdS[i] = analogRead(i); // Read all the CdS photocells
      Serial.print(CdS[i]); // Print the values to the console
    }
  }
}
```

August 12, 2015, SHB
Testing the Arduino Nano

Serial.println("");  
}
delay(200); // Wait a bit to keep the flickering to a minimum

// Blank the terminal so the data scrolling is legible
Serial.write(27); // ESC Character
Serial.println("[2J"); // Clear Screen
Serial.write(27); // ESC Character
Serial.println("[H"); // Cursor to Upper Left

}
Testing the Arduino Nano

The test process is as follows:

- Put the Arduino Nano in the test board socket, making sure to place the USB connector next to the etched USB icon on the cover, and not next to the two reset switches.
- Plug in the USB cable between the Nano and the computer, and select the correct board, processor and COM port in the Tools menu. Note the COM port number, since you will need it with the terminal emulator software, Putty, used to view the analog input values.
- Open the Arduino software, and load the Nano Test sketch – the code should match that above, and that shown in the screenshot below.

```cpp
// Initialize variables & constants for the program
int i; // Loop counter
int j; // Another loop counter
const int blue[] = {3, 5, 6, 9, 10, 11}; // PWM LEDs
int CdS[8]; // Array for CdS analog values

void setup() { // All DIO lines set to outputs
    pinMode(0, OUTPUT);
    pinMode(1, OUTPUT);
    pinMode(2, OUTPUT);
    pinMode(3, OUTPUT);
    pinMode(4, OUTPUT);
    pinMode(5, OUTPUT);
    pinMode(6, OUTPUT);
    pinMode(7, OUTPUT);
    pinMode(8, OUTPUT);
    pinMode(9, OUTPUT);
    pinMode(10, OUTPUT);
    pinMode(11, OUTPUT);
    pinMode(12, OUTPUT);
}
```

August 12, 2015, SHB
Testing the Arduino Nano

- From the Windows Start Menu, open the terminal emulator program Putty, which looks like the screenshot below:

![PuTTY Configuration](image.png)

- Configure Putty as follows:
  - Select Serial as the connection type
  - Verify that the Baud rate is 9600
  - Set the COM port to that noted above when configuring the Nano
  - Nothing else needs to be adjusted

- Don’t start the terminal session just yet – wait a bit. The software complains if you do things out of order, and you’ll have to restart the process. Your results should look like this, with COM5 being the COM port in this example:
At this point, download the Nano Test code to the Arduino, and let it start to run.

Back in the Putty configuration window, press the Open button to start the terminal emulation session. This will reset the Arduino, expected behavior, but since it just restarts the test code, that is just fine.

The terminal emulation window will open, but won’t do anything just yet. It’ll look like the screenshot below.
Testing the Arduino Nano

- At this point, the digital I/O lines will blink, D0-D13. Verify that they all blink.
- Next, the blue LEDs will fade in and fade out, one at a time. Verify that they do so.
- Once they finish, the analog port testing will begin, and six values will appear in the terminal emulation window. These are the analog input lines A0-A7. They’ll flicker a bit as they are updated.
- Cover the CdS photocells with your hand, and verify that the analog input values drop in response to the drop in light level.
The last things to test are the reset switches, and the two power supply display switches, although those are a bit redundant... the Arduino won’t run without power.