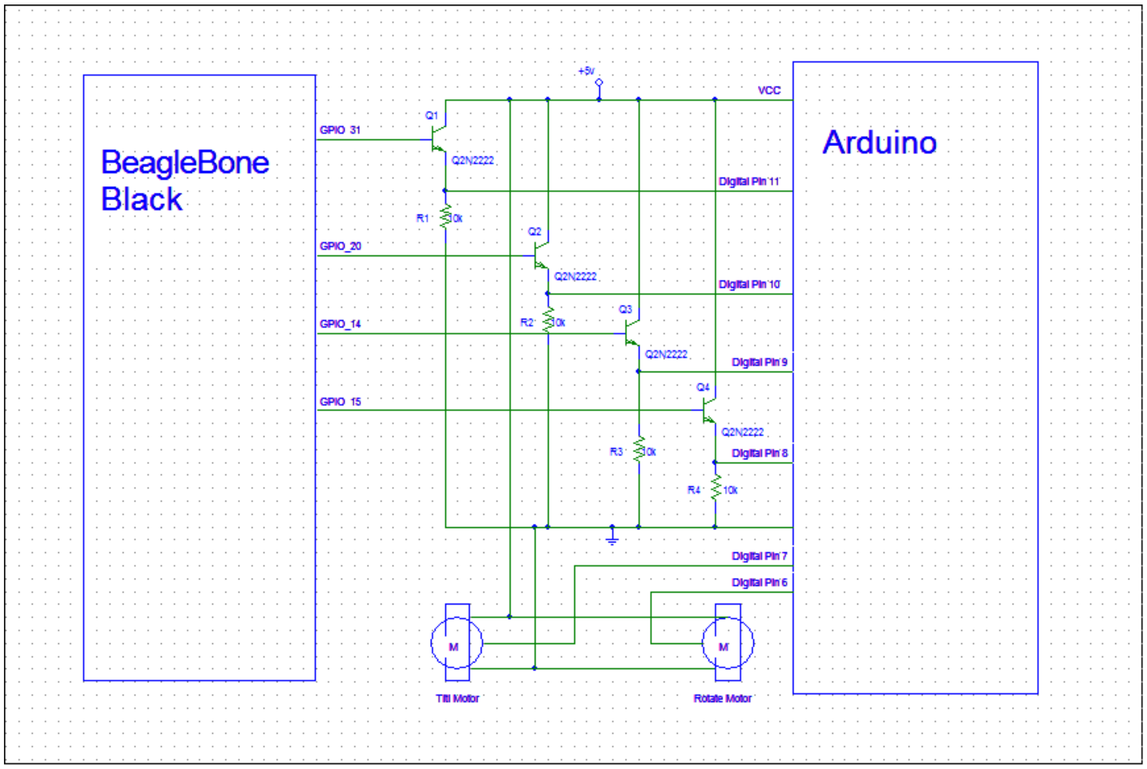
How to Guide

This is a guide of how to set up the BeagleBone Black to control servo motors using its GPIO pins. We use an Arduino microcontroller as an intermediate layer between the two. There will be 2 parts to this guide, one for setting up the hardware and one for setting up the software.

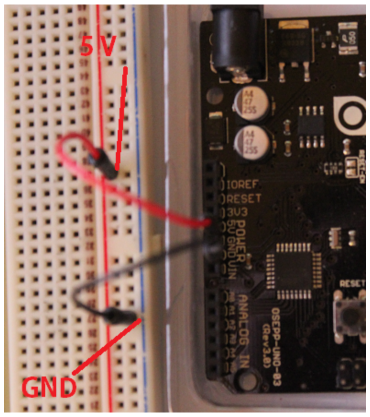
Hardware Set Up

For setting up the hardware, we'll be constructing the following schematic

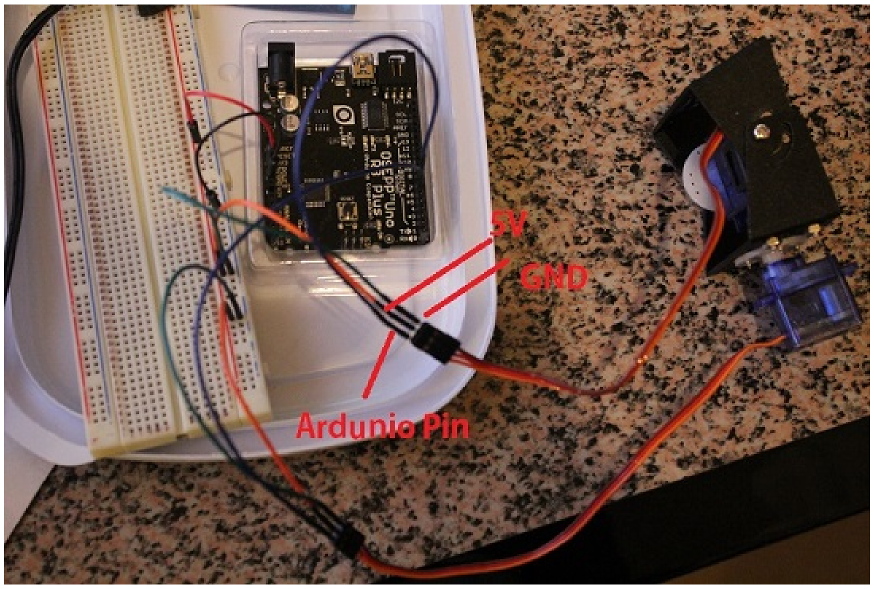


Steps:

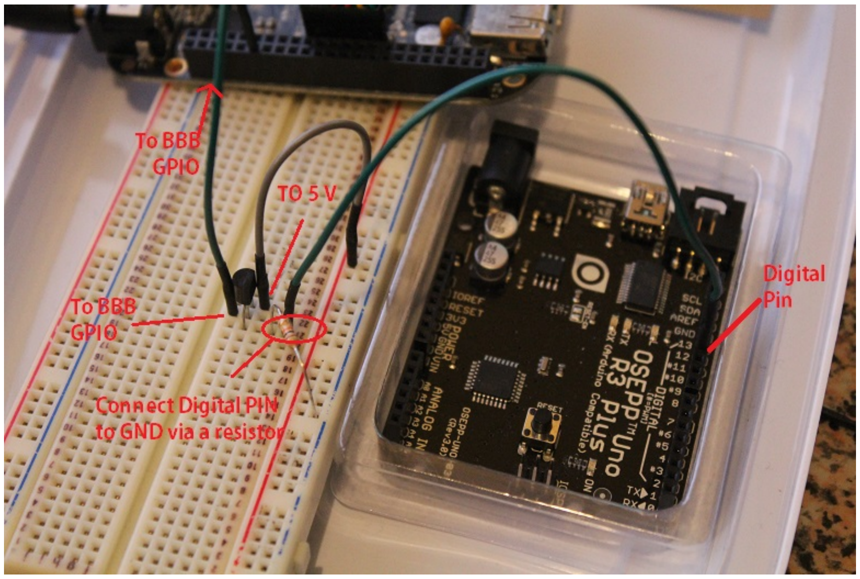
1. First wire the **5V** and **GND** from the Arduino board onto the bread beard board bus strips (the red and blue stripe on the side).

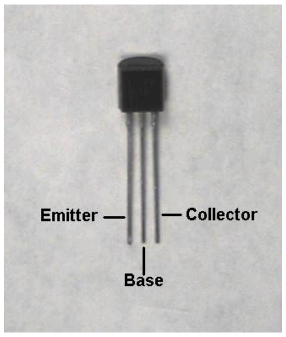


2. Next connect the servo motors to the breadboard (**5V** to the middle sockets, **GND** to the dark brown sockets, **Digital Pins 6 and 7** to each of the two remaining sockets)



3. Connect the BeagleBone **GPIO pins** to the Arduino **digital pins**. Use a pull down resistor to remove noise. To build a pull down resistor, do the following
    a. Place a transistor onto the breadboard so each pins is on it own terminal
    b. Place a jumper wire from the Arduino digital pins to the **emitter** pin of the transistor
    c. Place a **10k ohms resistor** connecting the **emitter** of the transistor to **GND**
    d. Use a jumper wire to connect the transistor's **collector** to **5V**
    e. Finally connect the **base** of the transistor to the BeagleBone **GPIO pin**

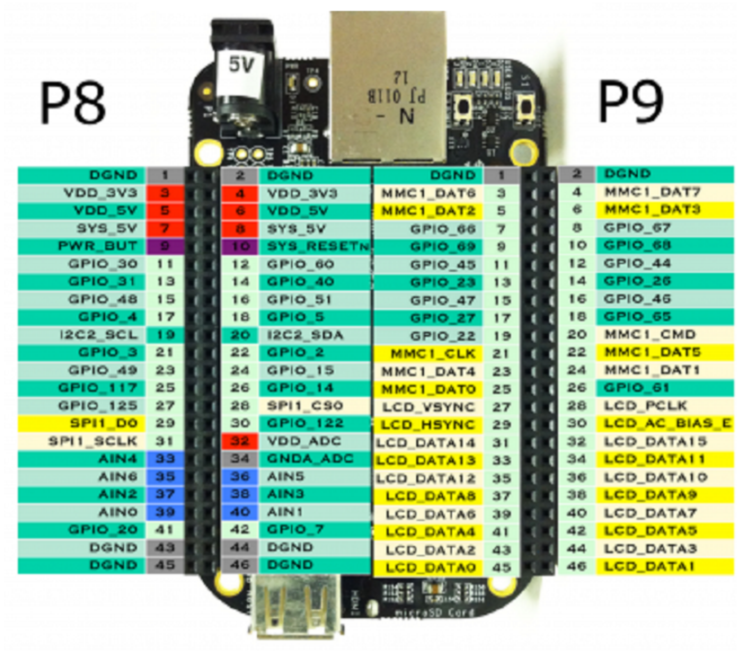




f. Do steps *a* to *e* for each pairs of BeagleBone GPIO and Arduino pins combination. They are as follows

BeagleBone GPIO pins	**Arduino Digital Pins**
GPIO_30	11
GPIO_20	10
GPIO_15	9
GPIO_14	8

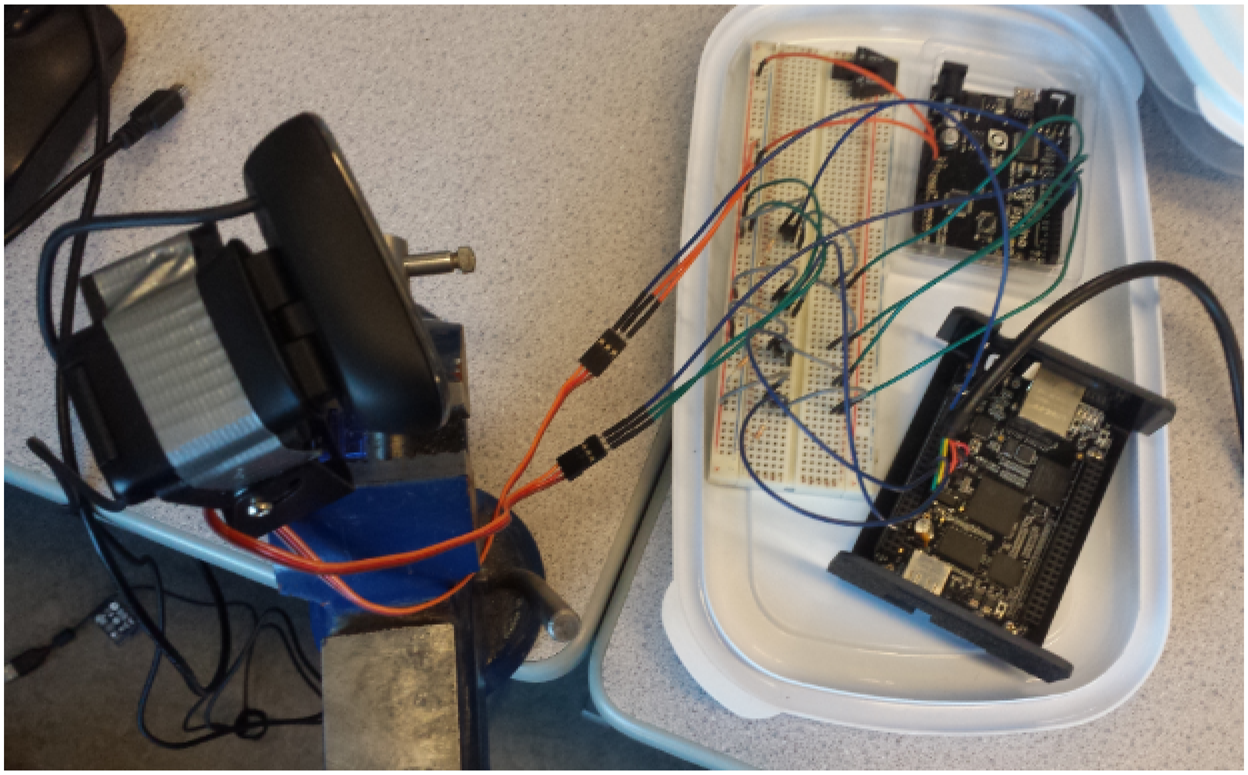
Pin Names for Beaglebone Black



P8			
DGND	1	2	DGND
VDD_3V3	3	4	VDD_3V3
VDD_5V	5	6	VDD_5V
SYS_5V	7	8	SYS_5V
PWR_BUT	9	10	SYS_RESETN
GPIO_30	11	12	GPIO_60
GPIO_31	13	14	GPIO_40
GPIO_48	15	16	GPIO_51
GPIO_4	17	18	GPIO_5
I2C2_SCL	19	20	I2C2_SDA
GPIO_3	21	22	GPIO_2
GPIO_49	23	24	GPIO_15
GPIO_117	25	26	GPIO_14
GPIO_125	27	28	SPI1_CS0
SPI1_D0	29	30	GPIO_122
SPI1_SCLK	31	32	VDD_ADC
AIN4	33	34	GNDA_ADC
AIN6	35	36	AIN5
AIN2	37	38	AIN3
AIN0	39	40	AIN1
GPIO_20	41	42	GPIO_7
DGND	43	44	DGND
DGND	45	46	DGND

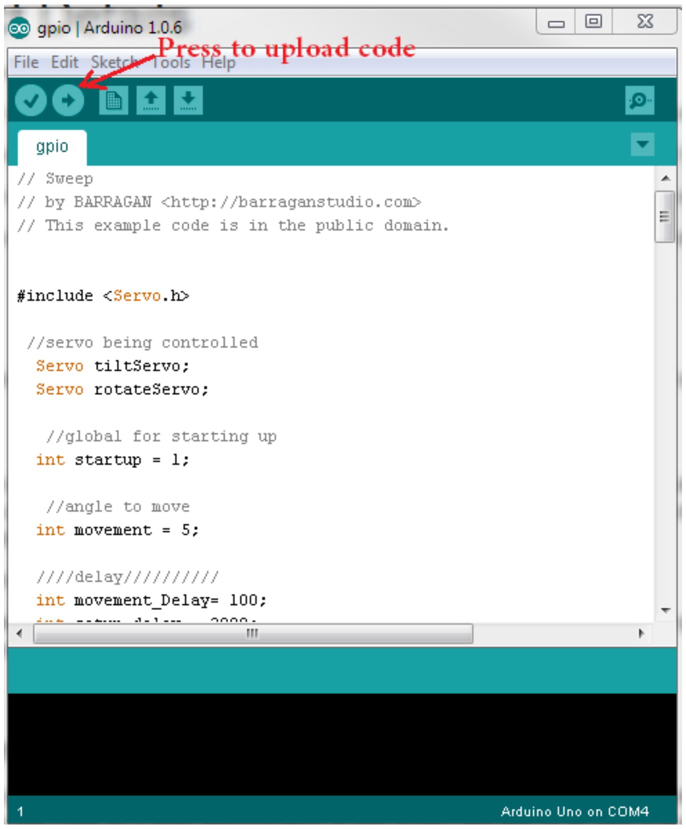
P9			
DGND	1	2	DGND
MMC1_DAT6	3	4	MMC1_DAT7
MMC1_DAT2	5	6	MMC1_DAT3
GPIO_66	7	8	GPIO_67
GPIO_69	9	10	GPIO_68
GPIO_45	11	12	GPIO_44
GPIO_23	13	14	GPIO_26
GPIO_47	15	16	GPIO_46
GPIO_27	17	18	GPIO_65
GPIO_22	19	20	MMC1_CMD
MMC1_CLK	21	22	MMC1_DAT5
MMC1_DAT4	23	24	MMC1_DAT1
MMC1_DAT0	25	26	GPIO_61
LCD_VSYNC	27	28	LCD_PCLK
LCD_HSYNC	29	30	LCD_AC_BIAS_E
LCD_DATA14	31	32	LCD_DATA15
LCD_DATA13	33	34	LCD_DATA11
LCD_DATA12	35	36	LCD_DATA10
LCD_DATA8	37	38	LCD_DATA9
LCD_DATA6	39	40	LCD_DATA7
LCD_DATA4	41	42	LCD_DATA5
LCD_DATA2	43	44	LCD_DATA3
LCD_DATA0	45	46	LCD_DATA1

4. Optionally attach a webcam or other device to the pan tilt and plug it into the BeagleBone usb port
5. Next attach the Arduino to a computer via USB port
6. The final set up would look like the following



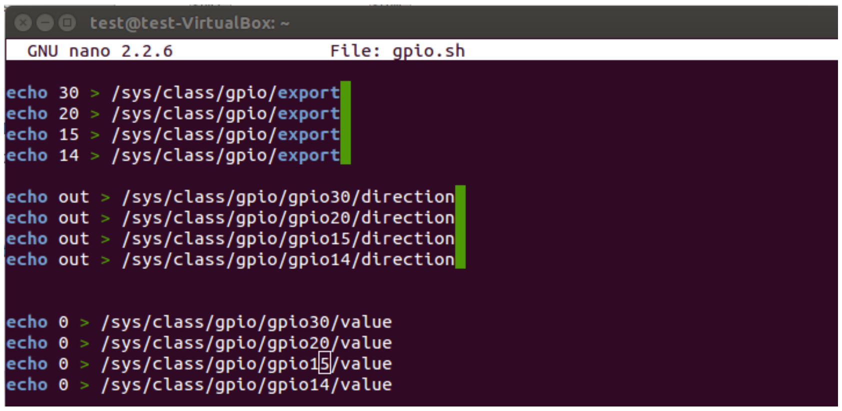
Software Setup

1. For setting up the Arduino, first open up the Arduino IDE
2. Load the “**gpio**” sketch into the IDE and press the “**Upload**” button to upload the code to the Arduino microcontroller
    a. **When the code is loaded, the Arduino will set both motors to an initial position of 90 degree. If it doesn’t do this then it’s not hooked up correctly**



```
// Sweep
// by BARRAGAN <http://barraganstudio.com>
// This example code is in the public domain.


#include <Servo.h>

 //servo being controlled
  Servo tiltServo;
  Servo rotateServo;

   //global for starting up
  int startup = 1;

   //angle to move
  int movement = 5;

  ////delay//////////
  int movement_Delay= 100;
```

3. Next start up the BeagleBone Black
4. On start up the BeagleBone Black will automatically run a "**gpio.sh**" script
    a. If the script isn't set up to run automatically, add the script to the user "**.profile**" file
    b. The script can also be run manually or the user could manually run the commands in the script one by one



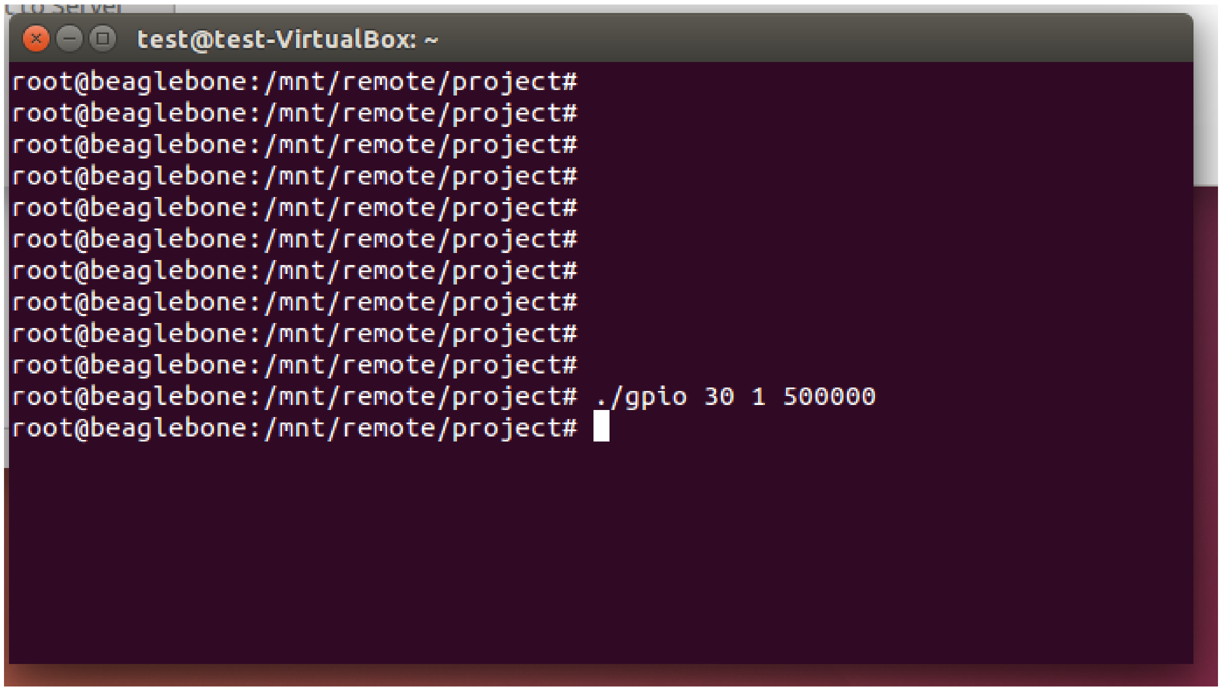
```
 GNU nano 2.2.6              File: gpio.sh

echo 30 > /sys/class/gpio/export
echo 20 > /sys/class/gpio/export
echo 15 > /sys/class/gpio/export
echo 14 > /sys/class/gpio/export

echo out > /sys/class/gpio/gpio30/direction
echo out > /sys/class/gpio/gpio20/direction
echo out > /sys/class/gpio/gpio15/direction
echo out > /sys/class/gpio/gpio14/direction


echo 0 > /sys/class/gpio/gpio30/value
echo 0 > /sys/class/gpio/gpio20/value
echo 0 > /sys/class/gpio/gpio15/value
echo 0 > /sys/class/gpio/gpio14/value
```

5. The GPIO pins are now ready to control the motors. Write a "**1**" to their "**value**" file to get the motors moving

a. The movement are as follows

GPIO Pins	Movement
GPIO_30	Rotate Left
GPIO_20	Rotate Right
GPIO_15	Tilt Up
GPIO_14	Tilt Down

6. Beside manually writing the GPIO controls, there is a “**gpio**” executable file written in C that control them.
    a. Run the “**gpio**” executable and give it a pin to set to it to high follow for a time for how long for it to remain high (seconds follow by nanoseconds)



```
root@beaglebone:/mnt/remote/project#
root@beaglebone:/mnt/remote/project#
root@beaglebone:/mnt/remote/project#
root@beaglebone:/mnt/remote/project#
root@beaglebone:/mnt/remote/project#
root@beaglebone:/mnt/remote/project#
root@beaglebone:/mnt/remote/project#
root@beaglebone:/mnt/remote/project#
root@beaglebone:/mnt/remote/project#
root@beaglebone:/mnt/remote/project#
root@beaglebone:/mnt/remote/project# ./gpio 30 1 500000
root@beaglebone:/mnt/remote/project#
```

Troubleshooting

The following is a troubleshooting guide for problem someone might encounter

Problem	**Solution**
Write “1” to GPIO output but nothing happens	Check that the motors are attach probably. Make sure that when the Arduino is loaded it moves the motors to 90 degrees
	For the Arduino IDE, open up the “**Serial Monitors**” under “**Tools**”. Set a GPIO pin to be 1, then connect it directly to a the input pins of the Arduino processor. See if the serial monitor shows that the Arduino is reading a 1. If it doesn’t read a 1, than the wire from the BeagleBone GPIO pin is in the wrong slot. Depending on the version of the BeagleBone Black and what layout diagram is used it is possible to have the wire in the wrong GPIO pin. Also try connecting the 3.3 V on the Arduino to itself to make sure it inputs is working. Feature more try checking the GPIO pins with a voltmeter or even the Arduino analog input pins
	Also try running a program that doesn’t nothing but move the motors. The official Arduino site has an example called “sweep”. One possible problem is that the motor do not have enough torque to move or they are stuck.
Everything is hooked up correctly but it doesn’t work	Make sure the BeagleBone Black is connected with the 5V adapter and not just using USB. The GPIO pins output 0-3.3 V, and if it connected only using USB it might not provide enough voltage to the transistor to reach it threshold voltage
	Check that all transistor are working, it is possible for them to be broken and not let voltage through even if it getting power to it base. Also make sure that they are facing the right direction.