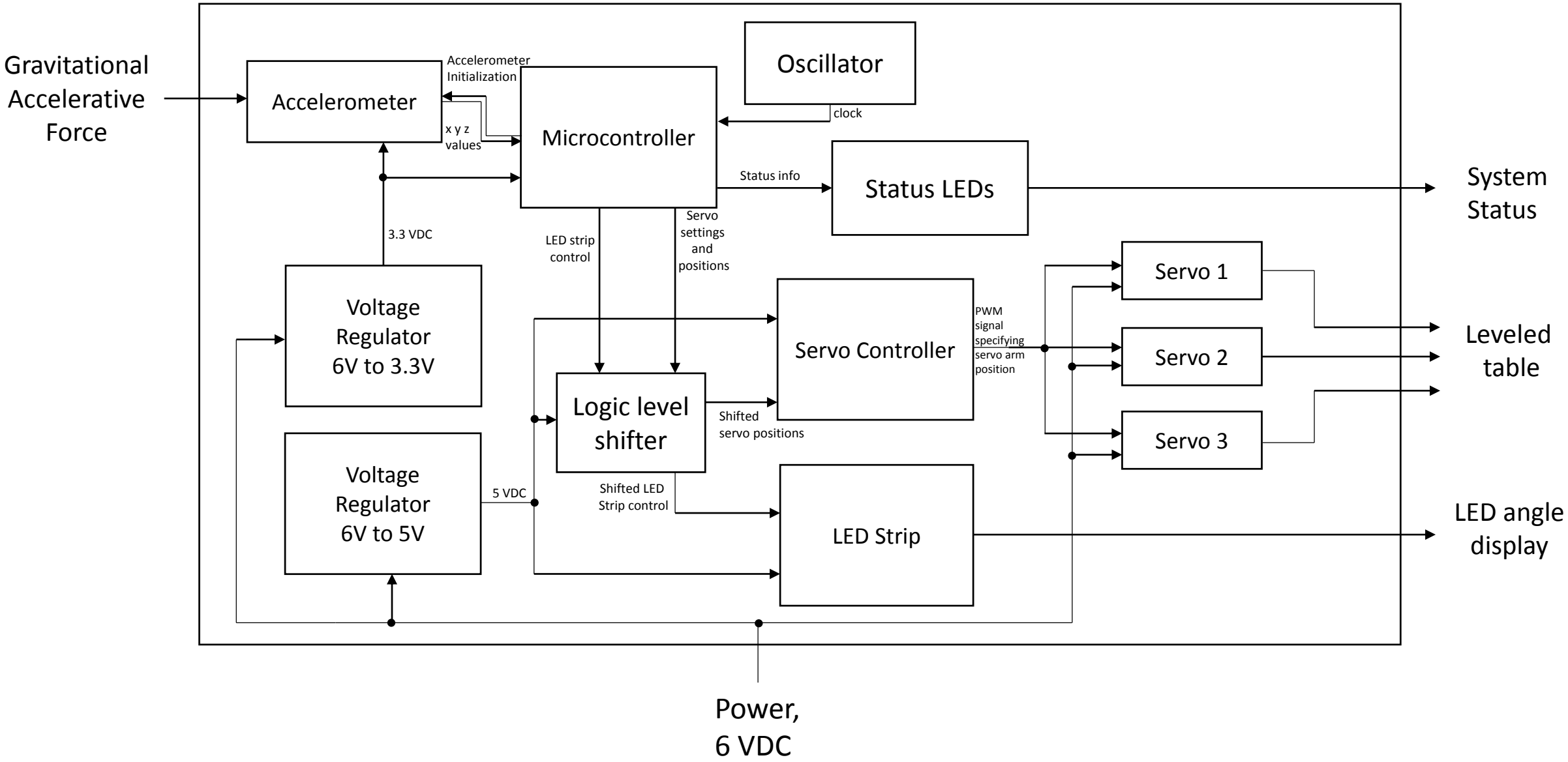
# T04: Self Leveling Table

Sean Hendrickson
Waleed Alhaddad
Adrian Steele
Taylor Griffin

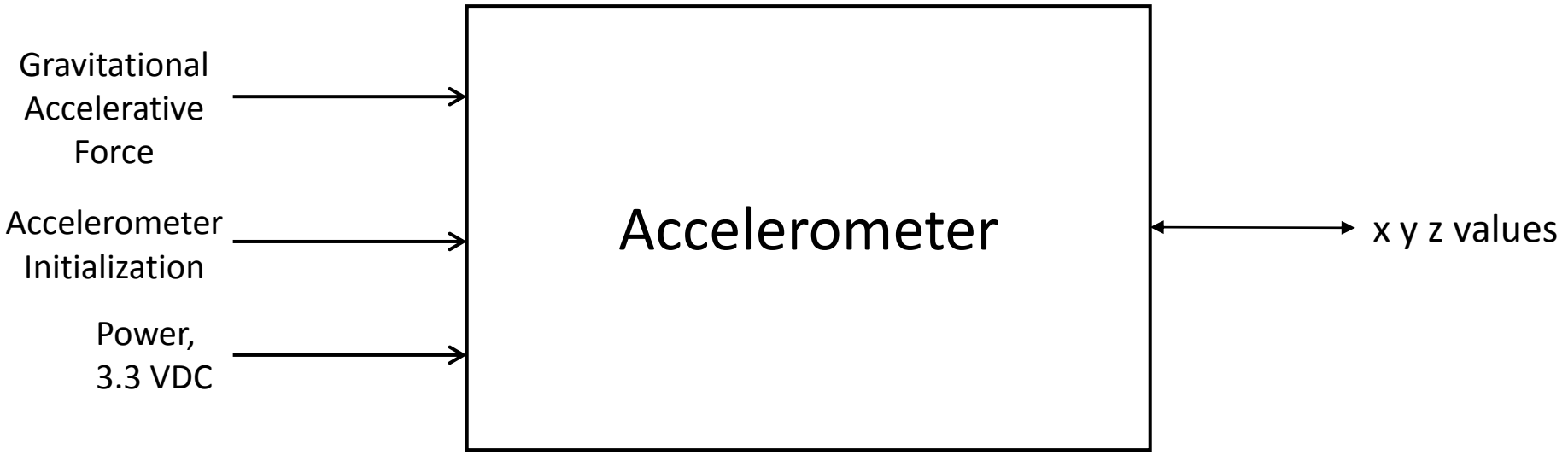
# Self Leveling Table: Level 0


| Module | Self Leveling Table |
|---|---|
| Inputs | - Gravitational Accelerative Force: The external forces acting on the device, used to determine the angle of the device and set servo position<br>- Power: 6 VDC from batteries. |
| Outputs | - LED Angle Display: 15 LEDs lit up in different configurations. Lights up based on device tilt slowly changing colors. LEDs around the device become more blue on the device side that is tilted up. LEDs around the device become more red on the device side that is tilted down.<br>- System status: 4 LEDs that light up on PCB. First LED is green, displays when tilt on the x axis reaches its limit. Second LED is yellow, displays when there is communication between the microcontroller and the servo controller. Third LED is yellow, displays when there is communication between the microcontroller and the accelerometer. Fourth LED is red, displays when the tilt on the y axis reaches its limit. |
| Functionality | - Consists of 2 layers: the base (two triangular acrylic sheets enclosing the batteries, PCB, servos, and LEDs), and the top tray (one triangular acrylic sheet leveled by servos). Receives input from an accelerometer and determines servo position to keep the top tray level. |

# Self Leveling Table Design: Level 1

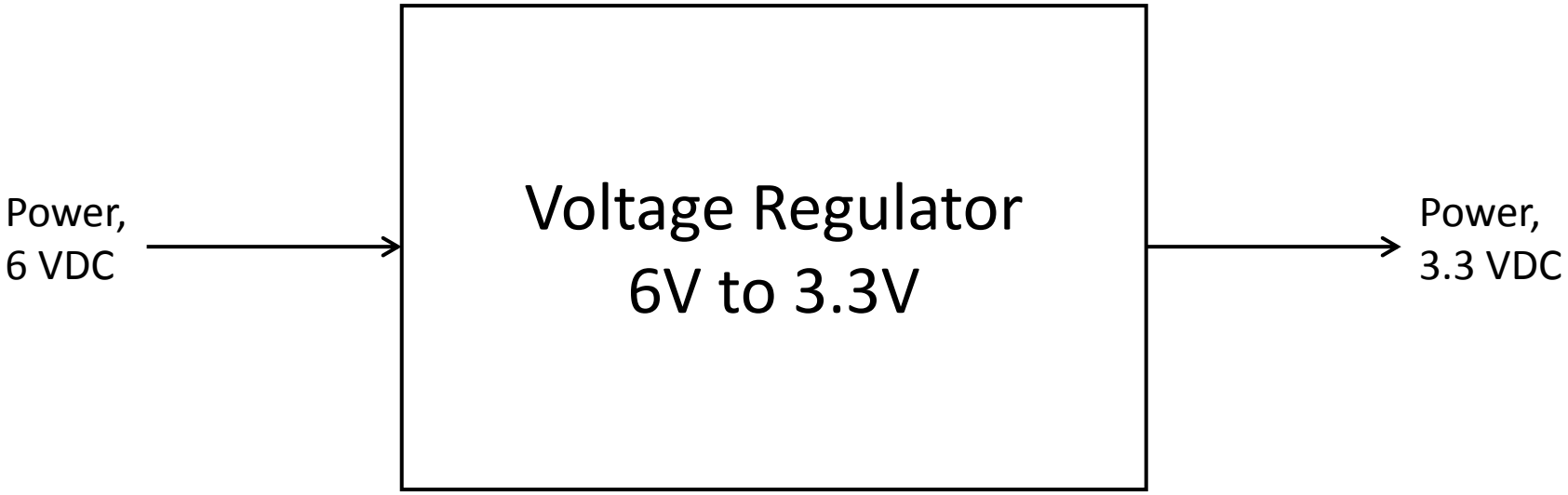


# Accelerometer: Level 0



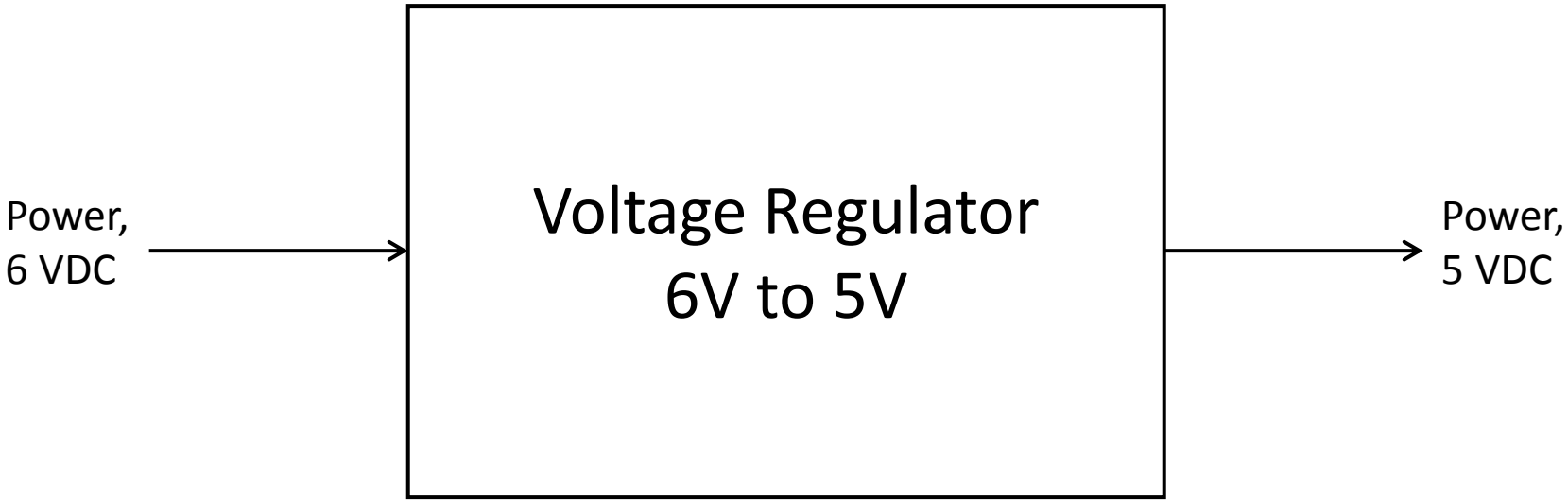

| Module | Accelerometer |
|---|---|
| Inputs | - Gravitational Accelerative Force: The accelerative force acting on the accelerometer<br>- Power: Regulated 3.3 VDC<br>- Accelerometer Initialization: Set 50 Hz sample rate, fast read mode (ignore last 4 data bits for x y z output), and activate output. Communicated via TWI. TWI format - send start condition, send accelerometer slave address (0x3A), send register to write to (0x23 - system control register 1), send value to put in register (0x23 - bit 5 = 1 bit 4 = 0 bit 3 = 0 data rate to 50 Hz, bit 1 = 1 fast read mode, bit 0 = 1 active mode), send stop condition. |
| Outputs | - x y z values: 8 most significant bits of 12, expressed in 2's complement, representing 1 of 256 values on a ± 2 g scale transmitted over TWI |
| Functionality | - Measures the accelerative forces, and outputs the values of the x y z directions. |

# Voltage Regulator 6V to 3.3V: Level 0



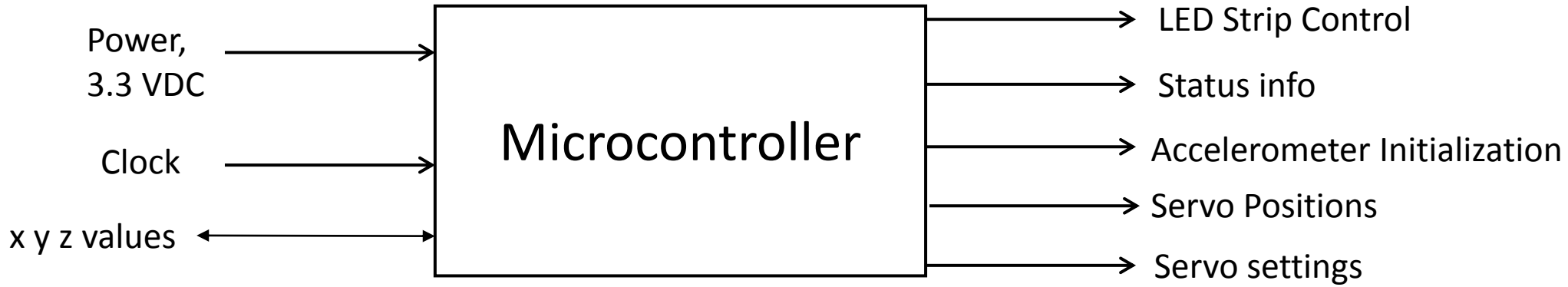

| Module | Voltage Regulator 6V to 3.3V |
|---|---|
| Inputs | - 6 V battery power supply |
| Outputs | - Regulated 3.3 VDC |
| Functionality | - Lowers the incoming 5-20 volts down to a regulated 3.3 VDC, outputting up to 500 mA |

# Voltage Regulator 6V to 5V: Level 0



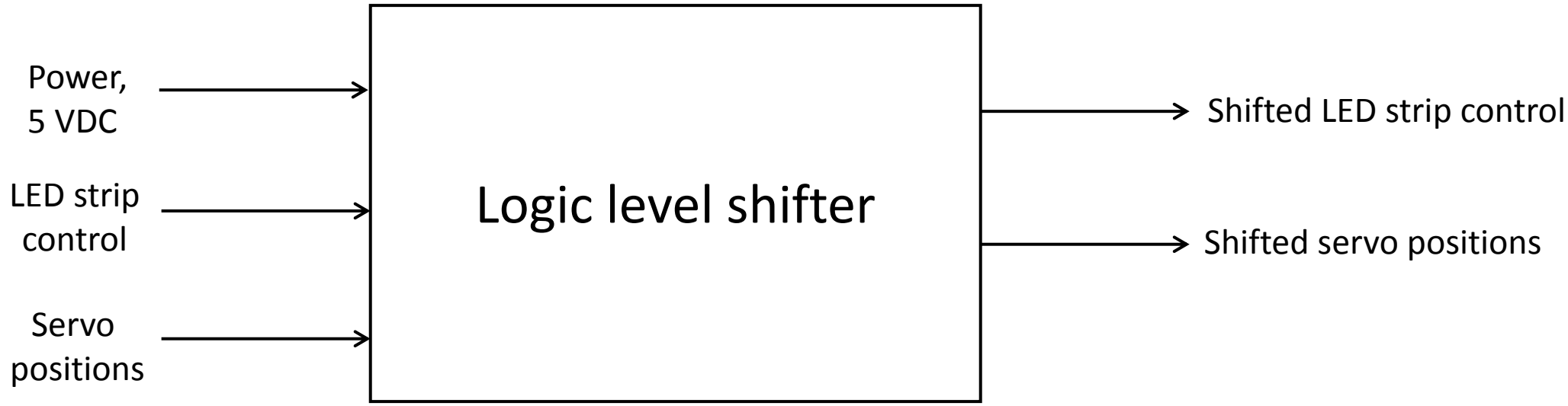

| Module | Voltage Regulator 6V to 5V |
|---|---|
| Inputs | - 6 V battery power supply |
| Outputs | - Regulated 5 VDC |
| Functionality | - Lowers the incoming 6-26 volts down to a regulated 5 VDC, outputting up to 1 A |

# Microcontroller: Level 0



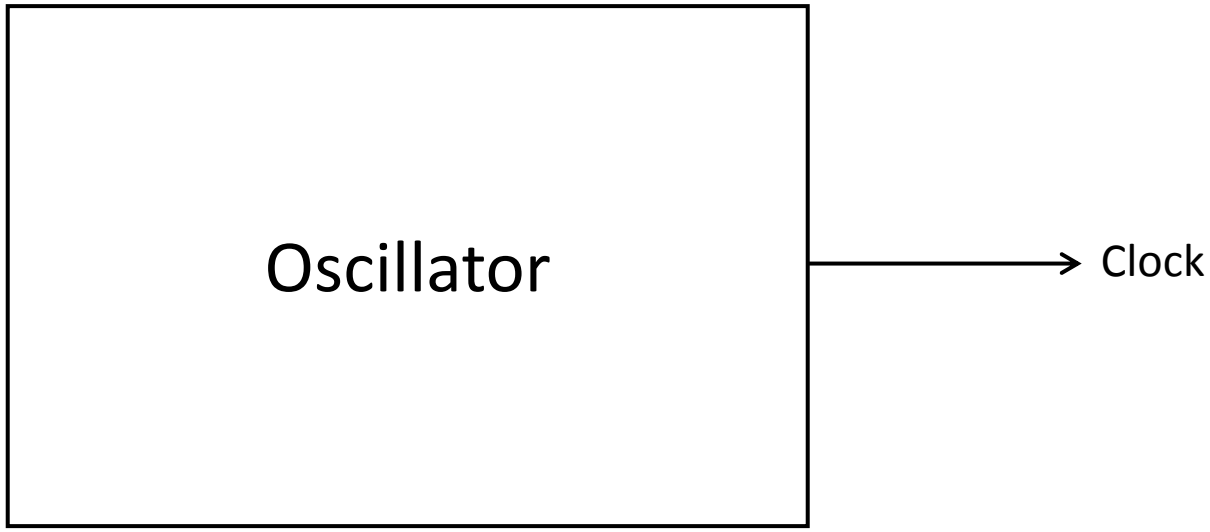

| Module | Microcontroller |
|---|---|
| Inputs | - Power: Regulated 3.3 VDC<br>- Clock: 8 MHz<br>- x y z input: 8 most significant bits of 12, expressed in 2's complement, representing 1 of 256 values on a ± 2 g scale transmitted over TWI |
| Outputs | - LED Strip control: which LEDs to light up. Communicated via SPI. Send 24 bit color pattern, shifted through the strip.<br>- Status info: 4 LEDs representing transmission maximum tilt statuses. Using 3.3 V GPIO. 1 green LED displaying when the tilt on the x axis has reached its limit. 2 yellow LEDs displaying when there is communication between the microcontroller and the accelerometer/servo controller. 1 Red LED displaying when the tilt on the y axis has reached its limit.<br>- Servo settings: Set the speed of the servos. speed limit is given in units of (0.25 µs)/(10 ms). Communicated via USART. Incoming communications can have varying baud rates but the ideal baud rate is 9600. Transmissions must follow the pololu protocol format and be transmitted one byte at a time (0xAA baud rate detection byte, 0x0C device ID of servo controller, 0x07 command byte to set servo speed, <channel number> channel of the servo that you want to set the speed, <servo speed MSB> least significant byte of speed value, <servo speed MSB> most significant byte of speed value)<br>- Servo positions: pulse width values in quarter µs (integer) and target servo to move. Communicated via USART. Incoming communications can have varying baud rates but the ideal baud rate is 9600. Transmissions must follow the pololu protocol format and be transmitted one byte at a time (0xAA baud rate detection byte, 0x0C device ID of servo controller, 0x04 command byte to set servo positions, <channel number> channel of the servo that you want to set the position of, <servo position LSB> least significant byte of PWM value, <servo position MSB> most significant byte of PWM value)<br>- Accelerometer Initialization: Set 50 Hz sample rate, fast read mode (ignore last 4 data bits for x y z output), and activate output. Communicated via TWI. TWI format - send start condition, send accelerometer slave address (0x3A), send register to write to (0x23 - system control register 1), send value to put in register (0x23 - bit 5 = 1 bit 4 = 0 bit 3 = 0 data rate to 50 Hz, bit 1 = 1 fast read mode, bit 0 = 1 active mode), send stop condition. |
| Functionality | - Converts x y z values to angles. Runs calculation of correct servo position based on input provided by accelerometer. Also sends LED strips configurations based on current servo positions. Displays system status via 4 LEDs on PCB |

# Logic level shifter: Level 0



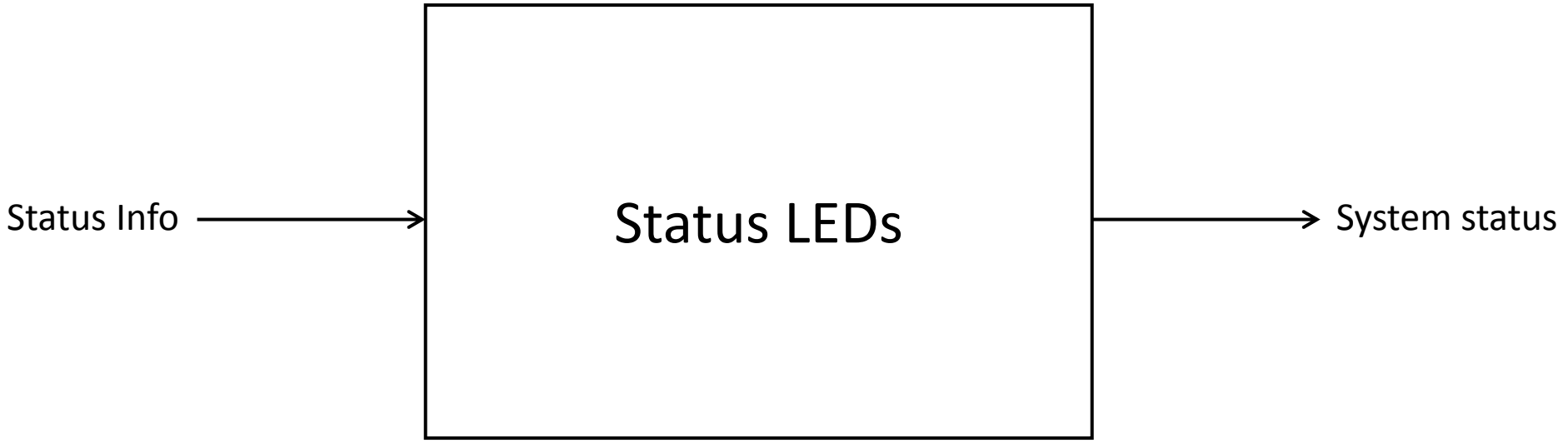

| Module | Logic Level Shifter |
|---|---|
| Inputs | - Power: Regulated 5 VDC<br>- LED Strip Control: LED display signals coming from microcontroller<br>- Servo position: servo position signals coming from microcontroller |
| Outputs | - Shifted LED Strip Control: Shifted LED signals from 3.3 to 5 V<br>- Shifted Servo position: Shifted servo position signals from 3.3 to 5 V |
| Functionality | - Shifts signals coming in from the microcontroller to a 5 V output |

# Oscillator: Level 0




| Module | Oscillator |
|---|---|
| Inputs | |
| Outputs | - Clock: 16 MHz |
| Functionality | - Sends the microcontroller an 8 MHz clock pulse |

# Status LEDs: Level 0

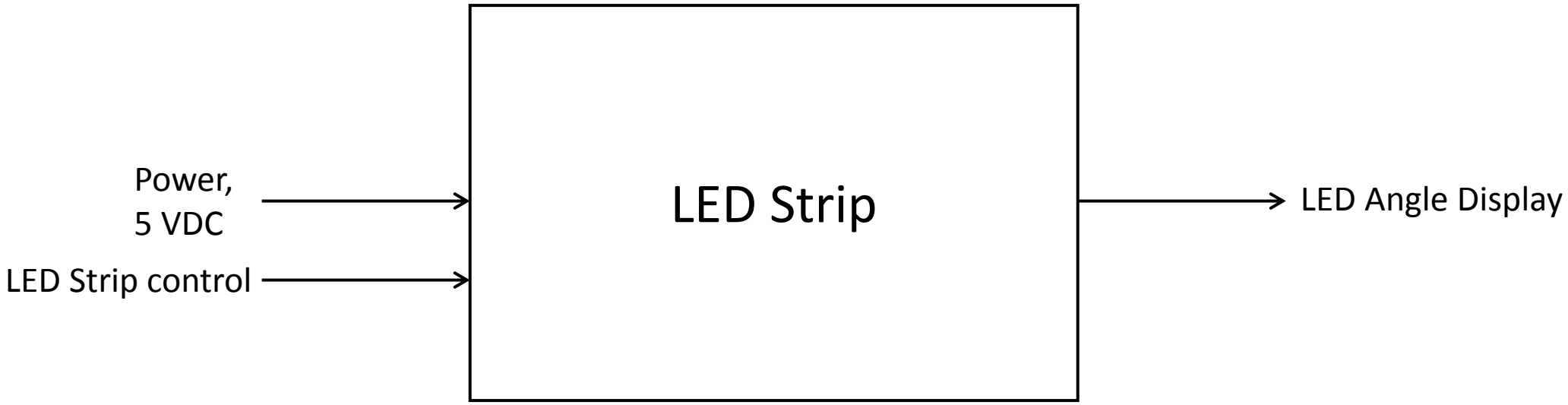



| Module | Status LEDs |
|---|---|
| Inputs | - Status Info: High or low signal depending on which LEDs to turn on. Communicated via 3.3 V GPIO |
| Outputs | - System status: 4 LEDs that light up on PCB. First LED is green, displays when tilt on the x axis reaches its limit. Second LED is yellow, displays when there is communication between the microcontroller and the servo controller. Third LED is yellow, displays when there is communication between the microcontroller and the accelerometer. Fourth LED is red, displays when the tilt on the y axis reaches its limit. |
| Functionality | - Displays status information including what transmissions are occurring (TWI, USART) and any errors |

# Servo Controller: Level 0

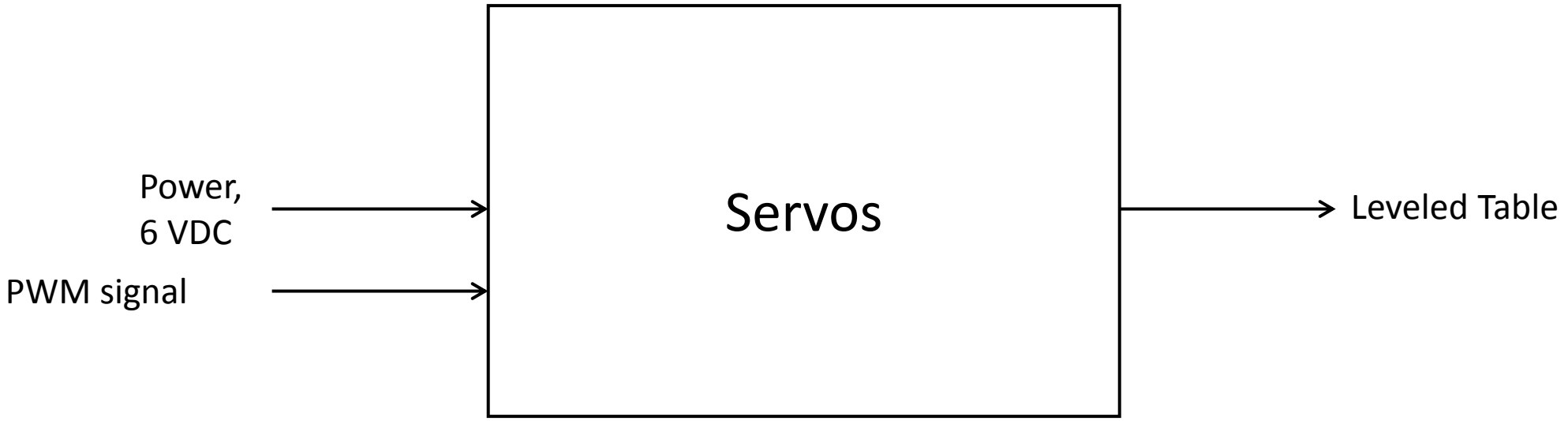

| Module | Servo Controller |
|---|---|
| Inputs | - Servo settings: Set the speed of the servos. speed limit is given in units of (0.25 µs)/(10 ms). Communicated via USART. Incoming communications can have varying baud rates but the ideal baud rate is 9600. Transmissions must follow the pololu protocol format and be transmitted one byte at a time (0xAA baud rate detection byte, 0x0C device ID of servo controller, 0x07 command byte to set servo speed, <channel number> channel of the servo that you want to set the speed, <servo speed MSB> least significant byte of speed value, <servo speed MSB> most significant byte of speed value)<br>- Power: Regulated 5 VDC<br>- Servo positions: pulse width values in quarter µs (integer) and target servo to move. Communicated via USART. Incoming communications can have varying baud rates but the ideal baud rate is 9600. Transmissions must follow the pololu protocol format and be transmitted one byte at a time (0xAA baud rate detection byte, 0x0C device ID of servo controller, 0x04 command byte to set servo positions, <channel number> channel of the servo that you want to set the position of, <servo position LSB> least significant byte of PWM value, <servo position MSB> most significant byte of PWM value) |
| Outputs | - Servo positions: Pulse width modulation signal to corresponding servo. Generates a pulse of a certain frequency to send to the servo what position it should hold. Done automatically, just need to receive input on what position to set the servo to. |
| Functionality | - Creating pulse width modulated signals to control up to 6 servos. |

# LED Strip: Level 0



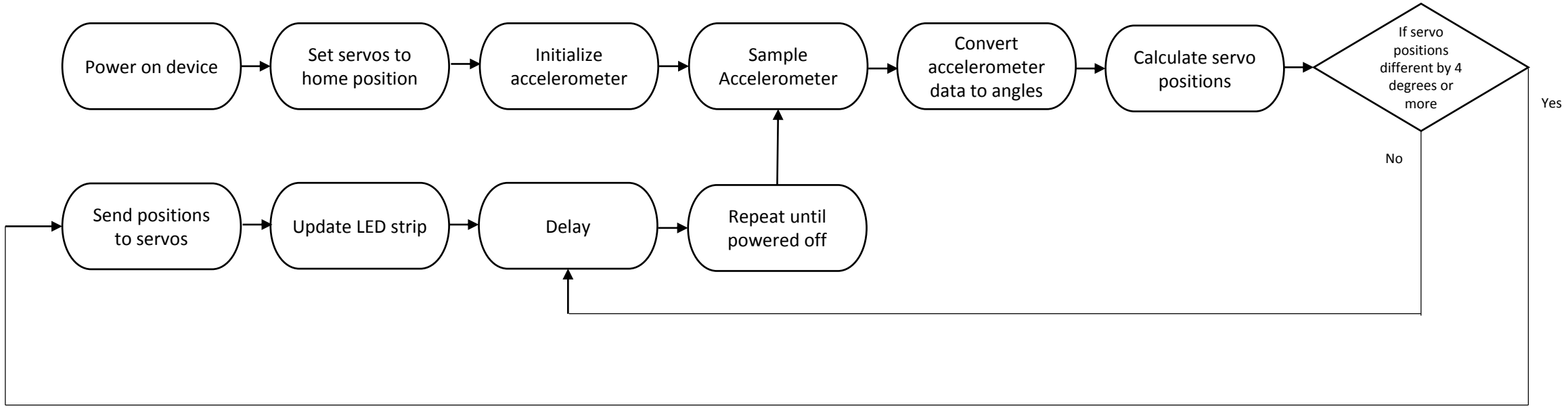

| Module | LED Strip |
|---|---|
| Inputs | - Power: Regulated 5 VDC<br>- LED Strip control: which LEDs to light up. Communicated via SPI. Send 24 bit color pattern, shifted through the strip. |
| Outputs | - Power: Regulated 5 VDC<br>- LED Strip control: which LEDs to light up. Communicated via SPI. Send 24 bit color pattern, shifted through the strip. |
| Functionality | - Decorative LED display based on each servo's position. Should not consume more than 1 A. Multicolored. |

# Servos: Level 0




| Module | LED Strip |
|---|---|
| Inputs | - - Power: 6 VDC<br>- - Servo positions: Pulse width modulation signal to corresponding servo. Generates a pulse of a certain frequency to send to the servo what position it should hold. Done automatically, just need to receive input on what position to set the servo to. |
| Outputs | - Leveled table: Adjust servo positions to level the top tray |
| Functionality | - The servo position is determined by the main program so that they keep the tray balanced. Range of 140° |

# UML: Activity Diagram




*Note: Update status LEDs after every communication and calculation and in the event of an error.

# UML: Interaction Diagram