**XBee Modules**

The Digi International XBee 802.15.4 module will handle the wireless data communication. The XBee transmitter module operates on a 2.4GHz radio frequency to transmit data to a receiver. The receiver communicates with the microcontroller mounted on the robot. The XBee module is compatible with our Lynxmotion SSC-32U servo controller.

**Arduino Uno Microcontroller**

The Arduino Uno Microcontroller has 14 digital input/output pins, 6 analog inputs, a 16 MHz quartz crystal, a USB connection, power jack, an ICSP header, and a reset button. As for its namesake, the Arduino Uno was the first to mark the release of the Arduino Software 1.0 IDE. The Uno has the ATmega328 preprogrammed with a bootloader that allows us to upload new code without the use of an external hardware programmer, communicating through the original STK500 protocol.

**SSC-32U Servo Controller**

The SSC-32U can control up to 32 servo channels as well as having eight analog I/O ports. Bidirectional communication with query commands, synchronized, or “group” moves. Group moves allow any combination of servos to begin and end motion simultaneously. Motion controls can be immediate response, speed controlled, timed motion, or a combination.

**Project Design**

For controlling our SSC-32 Servo Controller with the Arduino Uno, we must first connect these two boards together. The transmitting pin (TX) from the Arduino Uno will connect to the receiving pin (RX) of the SSC-32, as well as connecting the ground pins from each board to each other. With the boards connected, the Arduino will send serial commands to the SSC-32 to control the servos. For programming the SSC-32, the servo motor positions are in terms of numbers rather than degrees. The initial position (0 degrees) of the servo is the position 500 while the furthest position, which corresponds to 180 degrees, is the value 2500. An example of a line of code for the SSC-32 would be *#15 P1500 T500*, the # representing the pin number on the SSC-32, the P representing the position of the servo motor to rotate to, and T representing the time that the servo should take to rotate to the desired position. From the Arduino IDE, the commands are sent in serial format, such as *Serial.println(“#15 P1500 T500”);*. In order for the code to work, the SSC-32 baud rate has to be 9600. For some servos, the final position can be less than 2500, so we can lower it if the servomotor appears twitching when reaching the final position.