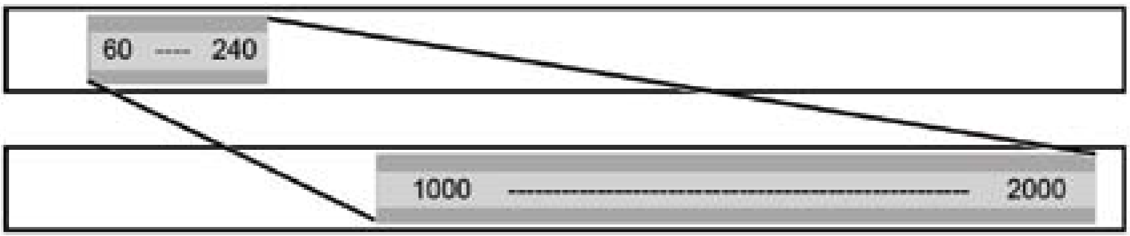
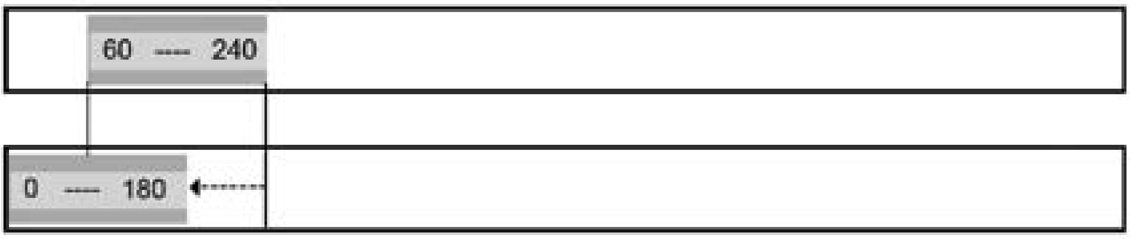
Scaling and transposing the numbers you get from the analog input:

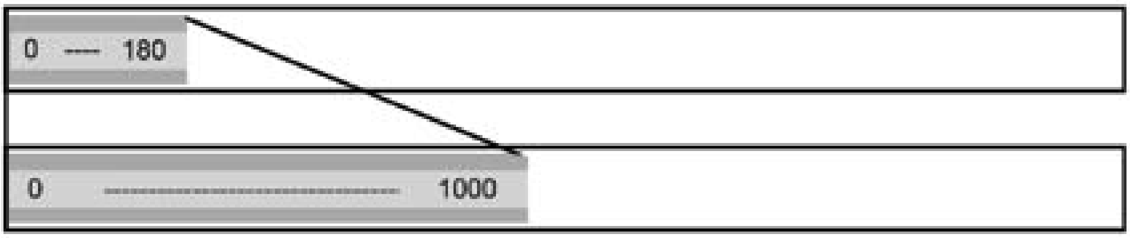
Let's say you want to take the range of numbers you are getting from the A-to-D conversion of an analog pin input and use it to position a servo. To get the full rotational range of the servo, we will be needing to translate this numerical range to a pulse width ranging from 1000-2000 microseconds. The analog input translates an infinite number of analog values between 0-5 volts into discreet integer steps between 0 and 255. But let's say that your sensor input isn't calibrated to fully exploit a 0-255 range. In our example the range turns out to be 60-240. How do we translate this range of numbers to our desired range of 1000-2000? Answer: Scale the width, then move it into the range that you want.



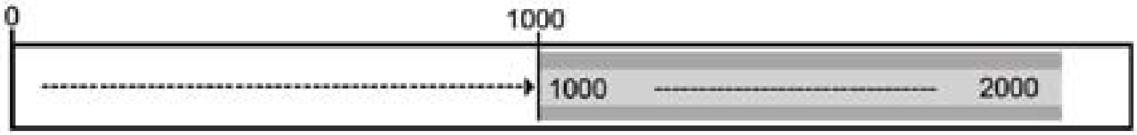
1st: **Subtract the difference between the lowest number and zero.** In our example, this means subtracting 60, effectively moving the range of numbers from 60-240 down to 0-180.



2nd: **Divide the desired high range number by the current highest number to find the scaling factor.** We will need to multiply our analog value by some number in order to scale the current range to the one we need. (e.g. *What do we need to multiply by so that our range goes from 0-1000 instead of 0-180?*) In our example we find this by dividing 1000/180. The result (5.55) is our *scaling factor*. Multiplying numbers between 0 and 180 by this scaling factor results in numbers between 0 and 1000.



3nd: **Add or subtract to move the range.** Now that we have the range correctly scaled, it is simply addition or subtraction to move these values to the range we need. (e.g. Adding 1000 would move the range of 0-1000 up to the range of 1000-2000.



So, in our example, we would take the numbers we are reading from the analog input and first multiply them by 5.55 and then displace them by the appropriate amount. (To move our 60-240 range up to a range beginning at 1000, this means adding 940.)

Code example:

```
r = adc_read(5);         // Read analog in on pin 5
r = (r * 5.55) + 940;    // scale and move range
```