NANOMUSCLE

Application Note

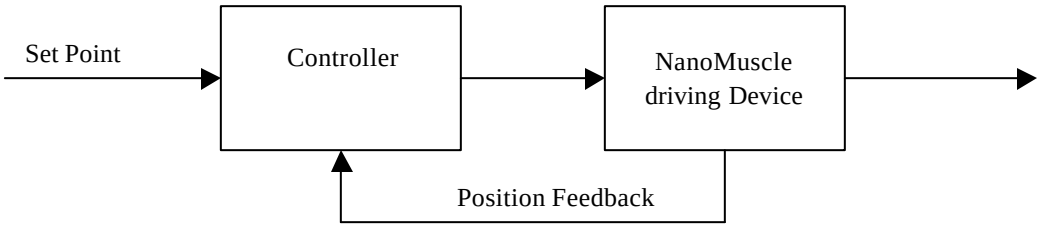
NM-AN04 – Position Control using NanoMuscle Actuators

Rev 1.2 – 19 March 2003

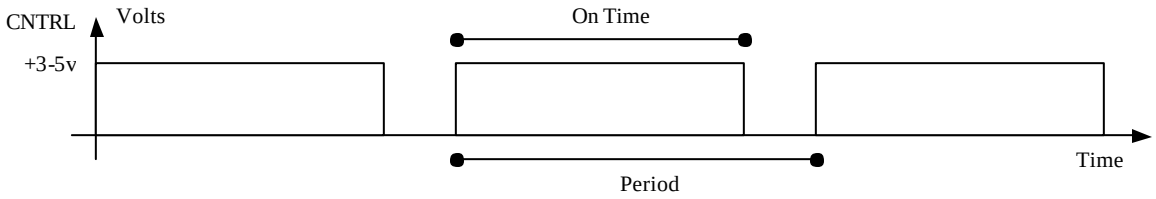
Applicable NanoMuscle Families: RS Rotary and NM Commercial

In their standard form all NanoMuscle actuators can provide simple full-range motion by simply having their CNTRL pin enabled. The NanoMuscle will keep contracting/rotating as long as the CNTRL pin is held high until it reaches its full extent. The rate at which this motion will occur will vary depending on, among other things, the load, friction in any linkage, voltage level and ambient temperature. Although for many applications a simple "on/off" approach is satisfactory, other applications require that the NanoMuscle be positioned at a given set point along its range of motion – in other words Position Control. This Application Note discusses the various options that can be used to achieve this.

Position control using a NanoMuscle is not fundamentally different to position control of any motor – you need a way of controlling the motor, some type of feedback sensor to measure the current position and a control process that knows the set point and controls the motor so that it homes in on this point.



For any effective position control, you need to be able to control the speed of movement of the NanoMuscle (as opposed to just full on or off). Since the control system is probably running on a micro-controller of some kind, the recommended way of controlling speed is using a Pulse Width Modulation (PWM) signal to the CNTRL pin rather than simply setting it . The speed of motion will be governed by the duty cycle of this PWM signal.



$$\text{Duty Cycle} = \frac{\text{On Time}}{\text{Period}}$$

The larger the duty cycle, the faster the NanoMuscle will move. Within certain constraints the actual period of the PWM does not have any affect. Unlike the control of some other devices, NanoMuscle actuators can operate at relatively low periods (typically to between 100 Hz and 1KHz). Many micro-controllers provide PWM support on-chip, so the generation of this signal is often as simple as loading the PWM circuit. However, due to the relatively low frequency it can also be driven in software.

Once speed is controlled, the second task is how to get feedback on the current position of the NanoMuscle. There are two ways in which the user can monitor the actual NanoMuscle:

Using external sensors	Options are optical encoders or some kind of resistive strip/potentiometer.
Using internal sensor	Some NanoMuscles contain an internal position sensor. This position sensor is analog, but with the right circuitry can be interpreted to obtain the position.

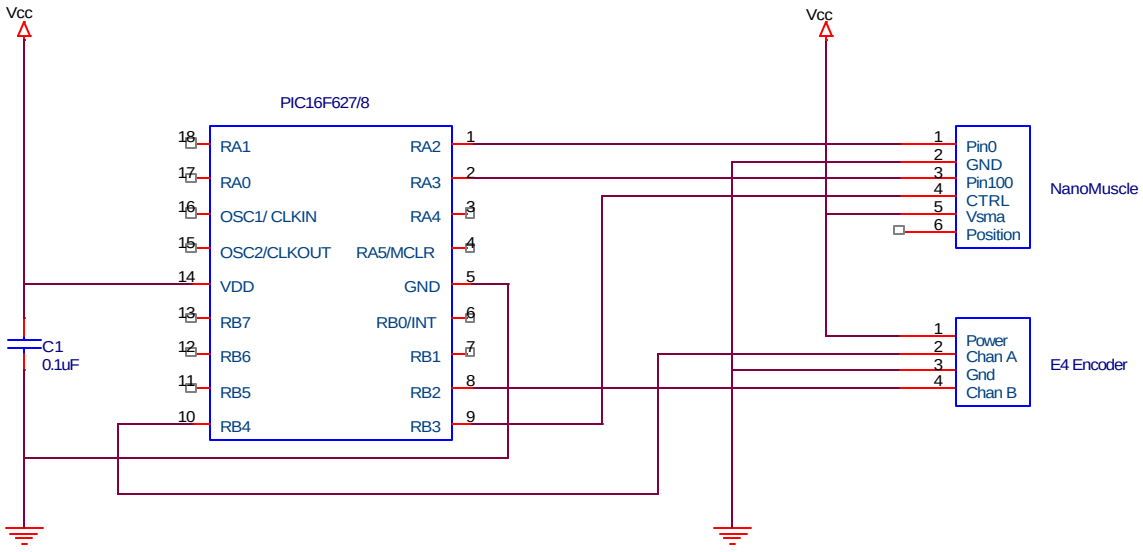
Typically two key factors affect the choice of sensor used: the required accuracy and cost. External sensors are available in many forms. Commercially available optical sensors can be accurate to +/- 0.5% or better of the full NanoMuscle travel, but are typically relatively expensive. Lower resolution optical encoders (e.g. 24-30 counts for the 60 degrees NanoMuscle rotation) can be built using off the shelf components – see the Application Note *NM-AN06 RS Low-cost Optical Encoder*. Resistive strips/pots can be relatively inexpensive depending on the accuracy required. The internal sensor can provide approximate position control with an accuracy of +/- 7%, but has the advantage of no additional hardware.

An example using Optical Sensors

This example uses a rotary optical encoder attached to an RS-125-CE rotary NanoMuscle. The sensor used is a US Digital E4-300-059 (see www.usdigital.com for more information).

This particular encoder comes with the correct shaft diameter and can be fitted directly onto the NanoMuscle. The output is a 2-channel quadrature signal that can be

fed directly into a micro-controller. In this example we have used the Microchip PIC 16F628 micro-controller. This chip has an onboard PWM circuit for driving the NanoMuscle and uses an interrupt pin on the A channel of the optical encoder. A typical schematic would be:



There are several stages in developing the software to control this system. First an interrupt handler will be needed that responds to the pulse train being received from the optical encoder. The quadrature signal returned has the advantage that direction of travel can be detected. In addition, the user can vary the accuracy of the count interpreted from a quadrature signal by how many edges are triggered/interrupted on. The following example shows use of a medium count level by being interrupted on both leading and trailing edge of Channel A only, using the lag/lead of Channel B to tell the direction of travel:

```
Interrupt_on_R4_change_sr()
{
      If Pin(R4) is High
      Then
            If (Pin(R3) is High
            Then
                  Position++;
            Else
                  Position--;
            Endif
      Else
            If (Pin(R3) is High
            Then
                  Position--;
            Else
                  Position++;
            Endif
      Endif
}
```

Refer to the documentation on the encoder for more information on quadrature signals.

Now that position of the NanoMuscle can be recorded, developing a control loop that uses this recorded position as well as the desired set point to drive the NanoMuscle must be done. Input of the set point will depend on the system in question; it might be manual via some physical hardware (e.g. slider/potentiometer) or derived from another digital system. The high level pseudo code for a control algorithm that adjusts the PWN duty cycle using feedback from the sensor is shown below:

```
For ever do
{
      Get current target set point position
      Get current feedback position

      Normalize target point and feedback position to a common range
      current_error = current position – target position

      new_duty_cycle = PID_Calculate(current_error);

      Set Duty Cyle = new_duty_cycle;
}
```

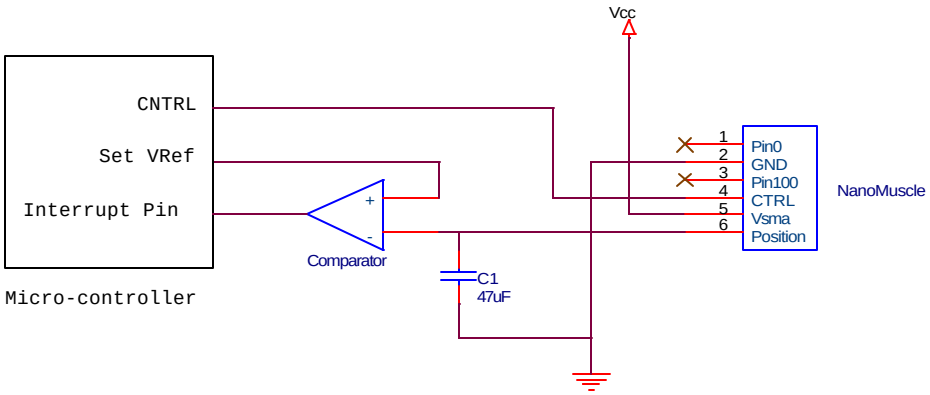
Depending on the tolerances in your system, the number of pulses for the complete range of travel for the NanoMuscle may or not be stable over time. If it is not, then an initial (or occasional) calibration sweep over the full range of motion may be needed to be able to normalize the returned encoder position.

The function PID_Calculate() will vary depending on the system in question. Like all control systems, the tuning of the constants used to calculate the Proportional, Integral and Derivative terms greatly affect the effectiveness of the response to a set point change.

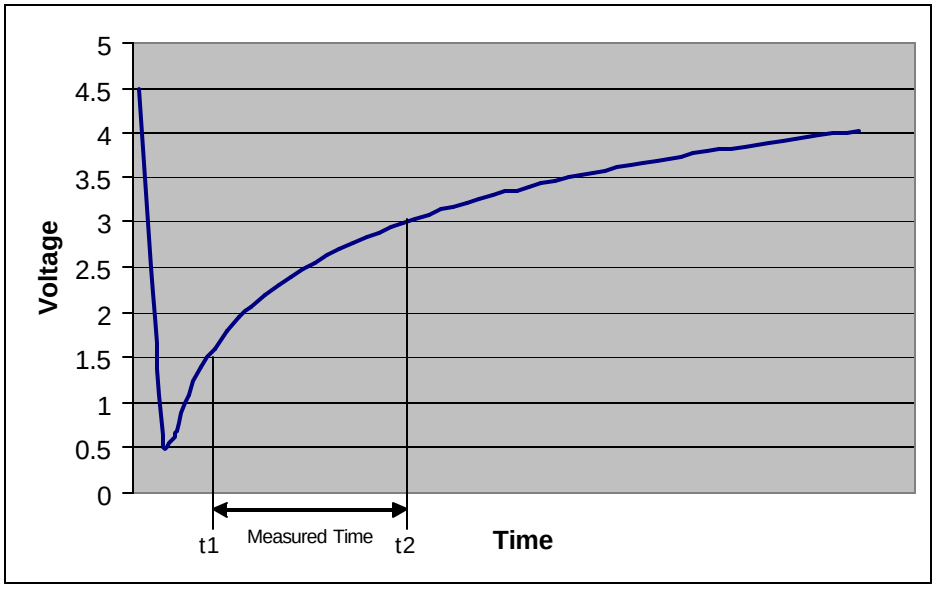
The above algorithms will work equally well for the NM Commercial NanoMuscles using a linear optical encoder. Likewise a resistive strip of some kind instead of the optical encoder could also be used.

An example using Internal Sensors

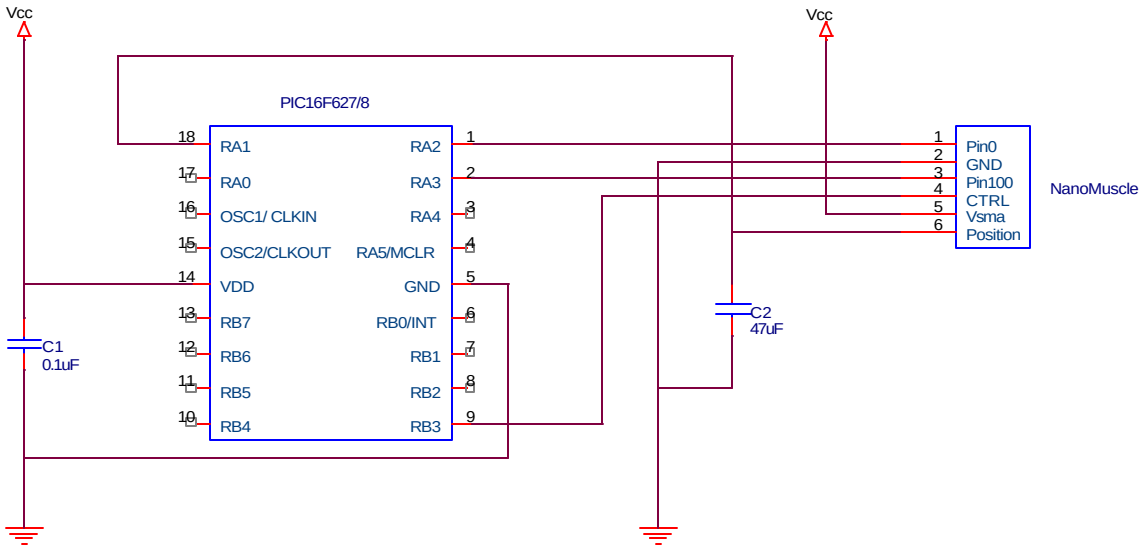
As mentioned earlier, some NanoMuscle actuators (namely the RS series) have a position feedback pin that gives an analog signal that can be interpreted to provide an approximate position. The change that must be measured is the change in resistance between the position feedback pin and the Vsma pin of the NanoMuscle. This resistance will vary around 10% during the contraction/rotation on the NanoMuscle. There are a number of methods that can be used to measure this resistance. Since the usual reason to use the internal sensor is to minimize cost, a simple solution is often employed. A second input into such design is that, for position control, an absolute measure of resistance is not required. Simply the ability to measure the relative resistance during the NanoMuscle movement is sufficient. A typical circuit used to measure this resistance is to time a capacitor charge up that is inserted in series with the resistance of the position control pin. Such a circuit is shown below:



Setting the CNTRL signal high of the NanoMuscle will drive the position feedback bin low and cause the capacitor to discharge. Clearing the CNTRL signal will start the capacitor charging. In order to be able to accurately time this charge up period, the above circuit uses a comparator with a Voltage Reference input that will trigger an interrupt in a micro-controller when the reference voltage is reached. Other solutions using Op-Amps or discrete components are also possible. The absolute resistance when un-contracted depends on the NanoMuscle in question, but is typically between 4 and 10 Ohms – hence the need for a fairly large capacitor in order to get a good charge up curve to measure (47uF is recommended). For best results it is recommended to actually time the period between two points on the charge up curve. This avoids problems with the spike at the start of the charge up curve that can vary between capacitors:



A full circuit to implement this resistance based position control is shown below, using the same micro-controller as in the previous example.



The PICF628 has the advantage that it has comparators (input on RA1 in the above example) and a voltage reference circuit on-chip; hence no additional components are required.

The code required for a resistance based position control system configured around this is very similar to that of the optical encoder. The only major difference is that instead of reading pulses from the encoder in an interrupt handler, the feedback resistance using the capacitor circuit must be periodically measured. The idealized pseudo code for such a reading is as follows:

```
GetResistance()
{
      Set Voltage Reference to low value (e.g. 0.8v)
      Set NanoMuscle CNTRL Pin High to discharge capacitor
      Wait for Comparator to indicate Voltage has dropped below VRef

      Set Voltage Reference to mid value (e.g. 1.5v)
      Set NanoMuscle CNTRL Pin Low to start capacitor charging
      Wait for Comparator to indicate Voltage has risen above VRef
      Start Timer

      Set Voltage Reference to high value (e.g. 3.5v)
      Wait for Comparator to indicate Voltage has risen above VRef
      Stop Timer

      RelativeResitance = Timer Value
}
```

Given an absolute resistance in the 4-8 Ohm range, the charge up time will be of the order (depending on the choice of mid and high VRef points) of 20-50 microseconds. Therefore, in practice, use an interrupt handler (triggered by the comparator signal) to start and stop the timer to ensure maximum accuracy. Additional items needing to be taken into account with such a position control system are:

- In order to be able to normalize the resistance readings, a calibration sweep is required.
- During any resistance measurement the application should not continue to drive the CNTRL pin with a PWM cycle since this will upset the measurement.
- Given the application is measuring relative small changes in resistance; there is likely to be some noise in the measurement. A simple averaging filter over a few readings is often beneficial in smoothing out such noise.

As mentioned earlier, internal resistance feedback can provide a good and inexpensive position control system with course grain accuracy – typically around +/- 7% of full NanoMuscle travel.

Developing your own position control system

To assist you in your own development, NanoMuscle has a range of Developer Kits, which contain a controller board supporting both the optical and resistance based control systems shown above. It also contains the same PIC micro-controller as used above and is provided with source code of the above algorithms. This software can be used as a basis for your own development. Details of the Development Kit can be found on the NanoMuscle web site www.nanomuscle.com (please note that the optical encoder itself is not included in the kit, but can be purchased direct from US Digital, www.usdigital.com).