This is a stepper motor controller using a 12c509 PIC and an octal driver to power the stepper motor. The motor is actuated by two push-buttons, one controls run-stop and the other controls the direction.
There is another implementation of this controller that uses an RS-232 interface. See here for details

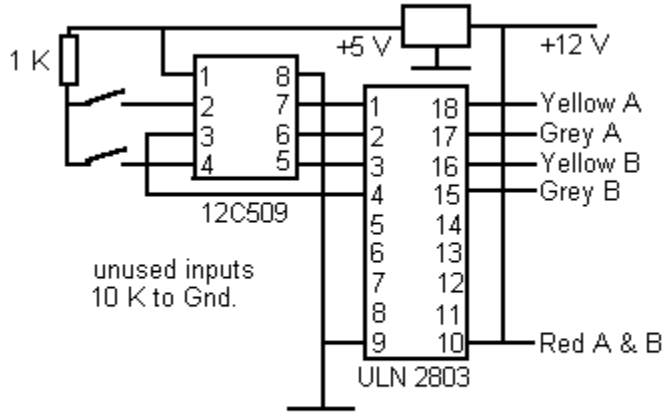
The controller actually runs the motor in half-steps. This means you get double the number of steps per revolution. A (nominally) 7.5 degree motor like the one I have that should give 48 steps per revolution here runs in steps of 3.75 degrees and gives 96 steps per revolution.

The motor is approx 2.5 cm. in diameter and takes roughly 120 mA when free-running.

The step time of the motor is 10mSec, and this can be changed inside the source of the controller code.

It should be possible to run **2** motors off the octal driver, each with their own PIC - or even running a more powerful motor by paralleling the unused drivers. Each of the 8 drivers can supply 500mA (with a 25% duty cycle, I am running a 50% duty cycle, be aware!)

Enough talk, first the circuit then the code



Stepper Motor Controller

O.k. some explainations
The **Yellow A & B** and **Grey A & B** refer to the colours of 4 of the 6 wires coming from the motor. I won't go into details about stepper motor design, there's lots of stuff on the web - some of it is even accurate!

The two **Red** wires are common, they go to the +12 Volt supply. Each of the 4 coils inside the motor are energised in sequence when a logic 1 is applied to one of the 4 inputs to the ULN 2803. This device contains 8 Darlington drivers which will allow current to pass through the coil and down to ground when the input is driven high.
The connection to pin 10 is for the flyback diodes inside the 2303.

By energising two of the coils at some times, the motor can be made to stop half-way between it's "normal" stopping points.

```
; control a small stepper motor with a 12c509
```

```
; There are two controls, run/stop and direction.
; The motor steps when the RUN button is depressed.
; If the DIRECTION button is pressed the direction
; of rotation is reversed.

;
; Peter Lynch, 13 May 1998
; pic@beowulf.demon.co.uk

; (this code is NOT certified Year 2000 compliant)

                LIST P=12c509

        include "\picde\P12c509.inc"

OPMASK  EQU     B'11000000'
BMASK   EQU     B'00101000'     ; all bits output (except GP3 and 5)
                                ; GP3 controls run/stop
                                ; GP5 controls direction

RUN_BTN EQU     3
DIR_BTN EQU     5

DELAY1  EQU     0C
DELAY2  EQU     0D

INDEX   EQU     0E              ; step index

        ORG 0

; start of main code

        MOVWF   OSCCAL

        MOVLW   OPMASK
        OPTION

        MOVLW   BMASK
        TRIS    GPIO


; now go into a loop, output the next bit pattern on
; GP0 - GP4 (GP3 is input only)

        CLRF    INDEX
NEXT

; check for the run/stop button

        BTFSS   GPIO, RUN_BTN
        GOTO    NEXT

        BTFSS   GPIO,DIR_BTN
        GOTO    CWISE

        INCF    INDEX, W
        GOTO    NEW_IDX
CWISE
        DECF    INDEX, W
NEW_IDX
        ANDLW   .7
        MOVWF   INDEX
```

```
; here W contains the index (either incremented or decremented, depending
; on the direction switch) into the array for the new stepper actuations

        CALL    STEP            ; convert the index into a bit pattern
        MOVWF   GPIO
        CALL    DELAY

        GOTO    NEXT


; routine to get step index

STEP
        ADDWF   PCL, F
        RETLW   B'00000001'
        RETLW   B'00000101'
        RETLW   B'00000100'
        RETLW   B'00000110'
        RETLW   B'00000010'
        RETLW   B'00010010'
        RETLW   B'00010000'
        RETLW   B'00010001'


; routine to delay between steps

DELAY

        MOVLW   .10             ; 10 mSec per step
        MOVWF   DELAY1
DEL_0
        MOVLW   .250            ; 1 millisecond delay
        MOVWF   DELAY2

DEL_1

        NOP
        DECFSZ  DELAY2, F
        GOTO    DEL_1

        DECFSZ  DELAY1, F
        GOTO    DEL_0

        RETURN

        END
```