Micro Stepper

How to build a microstepping controller using a Pic 16F628

Author : a.ratti

I have been thinking for a while, how to put togeter a simple microstepper controller without using a ready made commercial driver.
At the end, I came up with this simple project capable to drive unipolar stepper motor in three different modes

FULL STEP
HALF STEP
MICRO STEP

The Pic micro used in this work is a Pic 16F628 a powerfull small micro with PWM capability. The original idea was to obtain 256 micro steps, but soon I had to realize that 20 microsteps were the limit beyond which the spacing of microsteps were so unequal to become totaly useless and out of control. Nevertheless, I personally judge 1/20th of a step a very good result.

An additional improvement of this controller, is the capability to accept travel commands of 24 Bits (3 Bytes) that the controller put togheter and execute as positioning profile.
A travel commands with over 16 millions steps can keep the motor turns for several hours.

Controller accept commands via TTL serial port @ 9600,n,8,1 and return an (Ack) once the positioning profile has been completed.

HARDWARE:

As already mentioned the micro used is a Pic 16F628 plus 74HC86 (4 XOR gates) and 74HC04 (4 AND gates). The power output has been demanded to an ULN2803 with the transistors array used in parallel in order to increase the current to 1 Amp. **(See schematic)**

Due to a shortage of digital IO, the home switch has been used also as low limit switch. This means that when the motor is homed then it must move in the opposite direction, with enough steps to free the home switch , because during normal run this switch is seen by the system as a **LIMIT SWITCH.**

Home & Upper Limit Switch is the one tha the system will activate with the motor rotating endlessly in CW direction.

Home & Lower Limit Switch is the one that the system will activate with to motor rotating endlessly in CCW direction.

The above limit switches will act both as homing depending on how the homing function is activated. If homing CCW is activated than the UPPER LIMIT SWITCH will act as HOMING SWITCH. When the motor will reach the limit, motor will stop and move in the opposite direction (CW) for the number of steps declared in the 3 bytes travel value. So you not only activate the homing command, but will controll the home speed and the final home position.

On the contrary if a CW HOMING command is activated, than the motor will travel CW and the LOWER LIMIT SWITCH will act as HOMING SWITCH. Again, when motor has reached the limit switch, it will stop and move in the opposite direction (CCW) for the number of steps declared in the 3 bytes travel value.

Controller has 3 selectable modes to run the stepper motor :

Mode 1 **Full step double coils excitation**

Mode 2 **Half step**

Mode 3 **Micro Stepping (1/20th of a step)**

Once a mode is selected then it will act as a default mode till changed, and all motion commands (move & homing) will be executed in that mode. Changing mode could have effect on the true position of the system due to the change in the working coils sequence.

A working command is composed of a **Bytes Array** which is sent to the controller via TTL RS232. The array is seven bytes long with the following logical association:

Byte 0 **Address**

Byte 1 **Command**

Byte 2 **LSB** for motor velocity or **SYSTEM SETTING**

Byte 3 **MSB** for motor velocity

Byte 4 **LSB** for motor travel

Byte 5 **Middle Byte** for motor travel

Byte 6 **MSB** for motor travel

ADDRESS:

It can be any number from 0 to 255. By default the controller address is **10**

COMMANDS:

Thre are 10 different commands available and they are devided into two main groups. Motor commands and system commands. Are motor commands all the commands that have a motor movement as an answer. Are system commands all the commands that changes internal setting without moving the motors.

Motor Commands

1	Motor travel CW	Byte value = **60**
2	Motor travel CCW	Byte value = **50**
3	Homing CW	Byte value = **150**

4	Homing CCW	Byte value = **140**
5	Free motor	Byte value = **100**

System Commands

6	FullStep mode	Byte value = **210**
7	Half Step mode	Byte value = **220**
8	MicroStep mode	Byte value = **230**
9	Modify Ramp value	Byte value = **200**
10	Modify Address	Byte value = **190**

A valid command is so composed :

Address (1 byte) + Command (1 byte) + Velocity (2 Byte) + Travel (3 bytes)

When a command has been executed controller will acknowlege : **ascii 6 + command code + "ok"** if no failures has been detected during the run.

On the contrary if any failures will occur during the execution then controller will acknowlege : **ascii 7 + command code + "Fail"**

Examples:

The command : Serout [10,230,x,x,x,x,x] (where x = any number from 0 to 255)
will set controller to operate in MicroStep Mode

The command : Serout [10,200,88,2,x,x,x] (where x = any number from 0 to 255)
will set new Ramp value to 400 steps.

The command : Serout [10,150,88,2,32,3,0]
Will activate the Homing CCW. Motor will move till will reach Lower Limit Switch, then it will rotate CW for 800 steps (Mode=Last selected)

The command : Serout [10,140,88,2,144,1,0]
Will activate the Homing CW. Motor will move till will reach Higher Limit Switch, then it will rotate CCW for 400 steps (Mode=Last selected)

The command : Serout [10,60,88,2,32,3,0]
will turn the motor shaft CW for 800 steps with a delay of 600 (Mode=Last selected).

The command : Serout [10,50,88,2,32,3,0]
will turn the motor shaft CCW for 800 steps with a delay of 600 (Mode=Last selected).

If the motor will turn in the opposite direction respect to the command given, then swap coil wires

To familiarise with the command string, you can use the "Micro Stepping Service Tool" a VB program that will send to the controller commands already formatted.

Naturally you will need a serial level translator to connect the TTL RS232 to the PC serial port. (Maxim 232)

MOTOR VELOCITY

Velocity must be sent to the controller in units of delay, the number must be scomposed into two bytes. The low byte (LSB) and the high byte (MSB). Higher the number sent, slower the motor will turn. Slower the motor will turn, higher the torque developed by the motor shaft.

How to extract the Low Byte and the High Byte from the Ramp decimal value or velocity decimal value.

Assume for example we want to send **3000** units of delay than :

HighByte = Int(**3000** / 256) **(HighByte = 11)**

LowByte = **3000** – (HighByte * 256) **(LowByte = 184)**

MOTOR TRAVEL

Motor travel is given in steps. Minumum number permissible = 1 step (zero is ignored)
Maximum number = 16.5 millions steps
The total number of turn for a given number of steps will depend essentially from the type of motor.

A 1.8 degrees motor will make a complete turn with:
200 steps in FullStep Mode.
400 steps in HalfStep Mode
4000 steps in MicroStep Mode

A 0.9 degree motor will make a complete turn with:
400 steps in FullStep Mode
800 steps in HalfStep Mode
8000 steps in MicroStep Mode

The travel command is received by the controller as a three bytes value (byte4 + byte5 + byte6). The user must reduce his decimal number value, into a three separated bytes.

How to extract the three bytes from the travel value (Steps).

Let assume we want to send a command of 800,000 steps.

Int(800000/65536) = **12** (**HIGH BYTE**)

800000-(65536*12) = 13568

Int(13568/256) = **53** (**MID BYTE**)

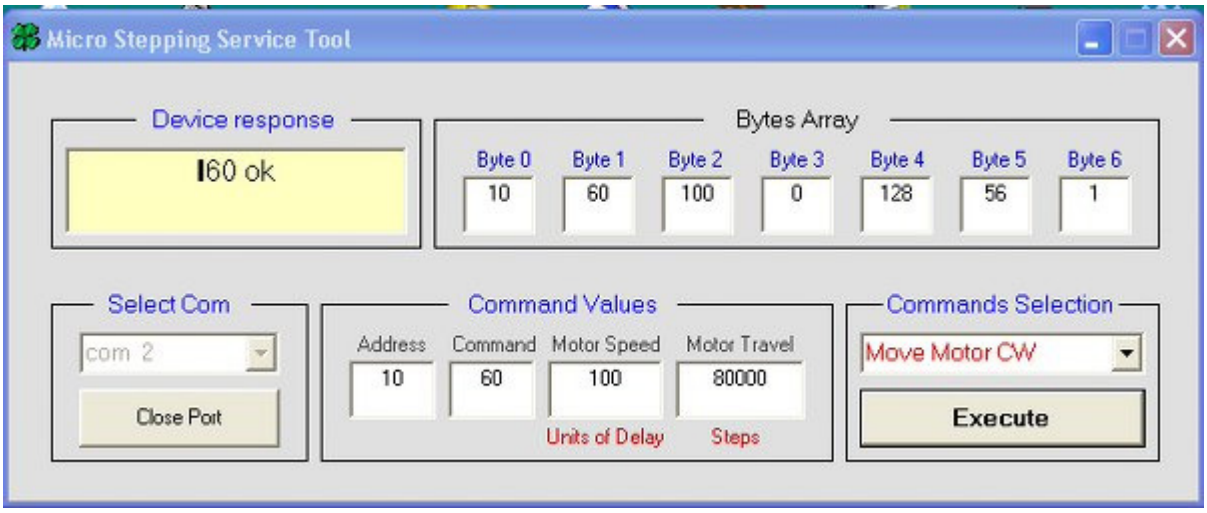
13568-(256*53) = **0** (**LOW BYTE**)

And, finally we have the 3 bytes to send to the controller.

Byte 4 = 0 (low byte)
Byte 5 = 53 (mid byte)
Byte 6 = 12 (high byte)

MICRO STEPPING SERVICE TOOL
As already mentioned, this service tool is a program witten in visual basic. It is very simple to use, just open the proper serial com port (where the controller is connected), type the velocity and the number of steps you want your motor moves and select the direction, using the combo selector. A this stage click "Execute" and the motor will turn.

In the box "Bytes Array" you will have the seven bytes command to obtain the motor move you have typed.

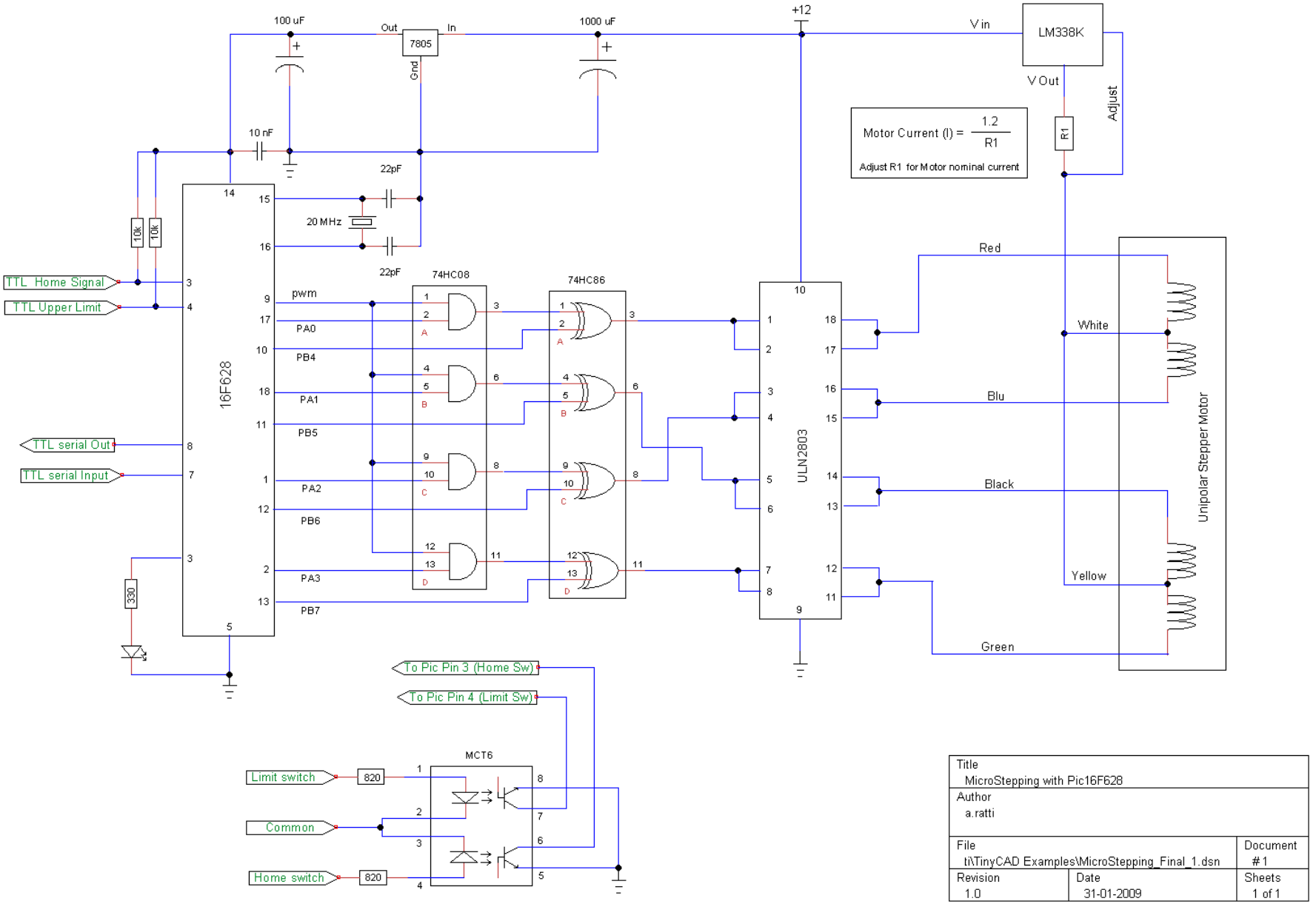


Remember that you will need a serial level translator to connect the TTL RS232 to the PC serial port. (for example a Maxim 232)

Unipolar motor used for the variuos tests:

VEXA stepping motor
2 PHASE
Model PK244M-02AA
DC 0.8 A.
Ohms = 7.5
Angle = 0.9
Wiring = see schematic

Software: Micro Stepping Service Tool.exe = VB dedicated tool application
MicroStepper_0.Hex = Pic hex code
(Remember to set Osc = HS and MCLR pin = input pin)



Title	MicroStepping with Pic16F628
Author	a.ratti
File	ti\TinyCAD Examples\MicroStepping_Final_1.dsn
Document	#1
Revision	1.0
Date	31-01-2009
Sheets	1 of 1