SmartMotor I/O Control at a Glance

Topics covered in this section: page

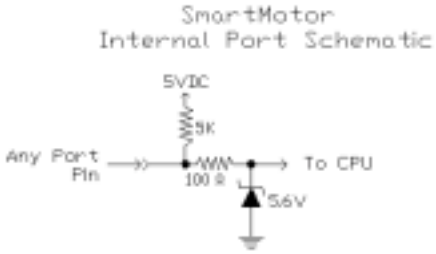
SmartMotor I/O Control at a Glance

There are seven I/O channels available on a Smart Motor.
They can be assigned as either Inputs, Outputs, or 10 Bit analog inputs individually.

Note: At any time, regardless of usage, the ports may be read as an analog value.

I/O Port Hardware

Each I/O point has a 100 Ohm series current limit resistor with a 5Kohm pull-up.
Each has a 5.6VDC zener diode as well.
Each can sink 15mA and source 4mA.



NOTE: They <u>are not</u> open collector P or N type outputs! They are Totem Pole CMOS driven outputs. When assigned as outputs they drive hard to either 5VDC or Ground when set to a 1 or 0 respectively

It is possible to use an Open Collector NPN type 24VDC prox with the SmartMotors.

The Use of PNP sourcing type 24VDC prox switches will damage the SmartMotor!

Dry contacts, if used, should be wired from the port pin to ground.
If you wire them from the port pin to 5VDC, the pull up resistor will cause you to always see a logic 1 on the port.

back to top

Software Control of I/O Ports

This is an overview of SmartMotor Code to control the on-board I/O:
In each case, X denotes any port A through G and should be replaced with the port letter to be used.

Reading a Port as an Input:

UXI : defines port "X" as an input port; X is any port A-F

Example:
UAI would define port "A" as an input port.

back to top

Software Control of I/O Ports (continued)

Assigning Port status to a variable:

x=UXI : assign the variable "x" the value at port "X".

: x will equal “1” if the port is at 5VDC.
: x will equal “0” if the port is at 0VDC.
: x can be any variable a-z, aa-zz, aaa-zzz..

Example: y=UAI would assign the logic state of port A to the variable y.

back to top

Reading a Port as an Analog value:

x=UXA : assigns the variable "x" a 10 Bit analog value from 0 to 1023 where 0 VDC=0 and 5VDC=1023
This command reads the selected port via a 10BIT A/D converter.

Example: z=UBA would assign the 10-Bit analog value of port B to the variable z.

Note: This is useful for self-diagnostics. By placing a known voltage dropping resistor on an external switch or other device, you can use the port as a general input, and yet do an analog read to check if the switch or sensor wire may have become disconnected from the port.

back to top

Assigning a Port as an Output:

UXO : defines port "X" as output port

Example: UEO would assign port B as an output.

UX=1 : drives voltage on port "X" to 5 VDC

Example: UE=1 would set port E to 5VDC.

UX=0 : drives voltage on port "X" to 0 VDC

Example: UE=0 would set port E to 0VDC.

Note: You can pre-define the state of a port prior to assigning it as an output.
Then you can toggle it between input and output to change from a driven state to a floating state. This is useful when tying I/O together between different SmartMotors.

back to top

Default States and special uses of I/O ports

Ports A and B Defaults and Specifics:

Ports A and B as quadrature encoder inputs:

Default state: Ports A and B default to phase A and B Encoder Inputs.
If no Port Control commands have been issued to Ports A or B, at any time, you can use the CTR command to get counter values from these ports:

Example: RCTR would report counter status from ports A and B

x=CTR would assign the counter value to the variable x

MF0 would re-set the counter to zero. (Mode Follow Zero)

Using MF1, MF2, MF4 or MFR will allow you to make use of ports A & B as quadrature encoder input ports and place the Motors into Electronic gearing or Mode Follow.
See Help files or manual for more.

Ports A and B as Step and Direction inputs:

The MSO command re-sets the counter to zero and sets up Ports A and B as Step and Direction inputs respectively. Once doing so, the same rules apply as listed above for the RCTR and CTR commands.

Setting up the motor as a stepper via MS, MS0, or MSR command will make port A be a step input and port B a direction input as well.
See Help files or manual for more.

Note: MS0 command is a good way of using port A as a high speed counter It zeros the Counter without changing the motor's mode of operation. From that point on, RCTR can report counts from prox switches, laser scanners etc.. as a high speed counter.

back to top

Default States and special uses of I/O ports (continued)

Ports C and D Defaults and Specifics:

Ports C and D default to right and left limit inputs respectively. If they are assigned as input or output ports, they must be redefined as limit switch inputs if you want them to be used as such. To redefine them as limit inputs you do the following:

Example: UCP redefines port "C" as right limit input (P for plus rotation)
UDM redefines port "D" as left limit input (M for minus rotation)

Ports E and F Defaults and Specifics:

Ports E and F can be used as the Anilink port or an RS-485 port. Any command used for communication to Anilink
devices will automatically set the ports as needed for Address and Data information.
Using the OCHN and OCHR commands with regards to RS-485 usage will automatically set the ports to half duplex RS-485 operation.
See Help files or manual for more.

Ports G Defaults and Specifics:

By Default, when port G sees a transition from 5 to zero volts, it means the same as typing a "G" and pressing enter or seeing a G in a program. This is why it is called the G Sync pin.
This allows you to synchronize “goes” on multiple motors at the same time.
The G Port can be redefined as general I/O like the other ports.
UG : command returns Port "G" to default sync port so if you assigned it as an input or output, To redefine it as "G sync" simply invoke the UG command again.

Port G also has the option of being used as a handshake line for RS-232 to RS-485 Adapters.
This is covered in the Help Files and manual under communications.

back to top

I/O Programming examples

The following are examples of triggering events off of Port I/O state changes.

Level Triggered Subroutine call

This code causes subroutine 100 to be called when port A goes high

```
IF UAI==1
        GOSUB100
ENDIF
```

Positive-Edge-Triggered Subroutine Call

This code causes subroutine 100 to be called when port A goes high only after first going low (negative pulse triggered subroutine call)

```
IF UAI==0
        WHILE UAI==0
        LOOP
        GOSUB100
ENDIF
```

Negative-Edge Triggered Subroutine Call

This code causes subroutine 100 to be called when port A goes low only after first going high (positive pulse triggered subroutine call)

```
IF UAI==1
        WHILE UAI==1
        LOOP
        GOSUB100
ENDIF
```

Level and State Change Print-Out example

```
WHILE 1==1              'while forever

        WHILE UBI==1 LOOP    'while port B is high, do nothing

        PRINT("Port B just went Low",#13)
        PRINT("Do nothing while it is Low",#13)

        WHILE UBI==0 LOOP    'while port B is low, do nothing

        PRINT("Port B just went High",#13)
        PRINT("Do nothing while it is High",#13)
LOOP
```

back to top