

# Motor Variations

## *Supported motor types*

Wed, Aug 2, 1995

Several types of motor interfaces are supported by the local station/IRM system. This document describes these possibilities and how to specify and use them.

### *Analog control field*

The primary motor parameters are set in the analog control field of the analog descriptor. Analog control type 0x02 is used for this purpose; hence, the 4-byte analog control field looks like this:

02	aux	ioAddr
----	-----	--------

The aux byte and the 2-byte ioAddr word are used in several ways for different motor interfaces.

### *CW-CCW memory-mapped interface*

This is a simple memory-mapped digital interface that consists of a pair of bits such that one delivers clockwise pulses and the other delivers counter-clockwise pulses, at a fixed rate of 150 Hz with 20  $\mu$ s pulse width. The pulse width allows for heavy filtering to eliminate noise, while the 150 Hz pulse rate is adequate for all motors actually in use in the Fermilab Linac control system. The ioAddr word is prefixed with a word of 0xFFFF to produce the full 32-bit I/O address that addresses the byte that contains the bit pair, allowing access to the entire VMEbus I/O space. Here, a "bit pair" means any pair of bits 7-6, 5-4, 3-2, or 1-0, where the most significant bit of the byte is bit 7. (One can't use bits 2-1 for a motor bit pair interface, for example.) For these cases, values of aux are between 0x00–0x07.

A variation for the CW-CCW interface allows using the value 0x10FF for the upper word of the full 32-bit I/O address. This allows access to a single auxiliary VME crate accessed via the vertical interconnect interface to crate 0x10 VME standard I/O. In this case, values of aux are 0x08–0x0F. The indicated bit# specified by the low 3 bits of the aux byte is the CW bit; the other bit of the pair is the CCW bit. If the indicated bit# is odd, then the pulses delivered are high-active; if the indicated bit# is even, the pulses are low-active. This means that the order to be used in wiring the bit pair depends upon whether the pulses are high-active or low-active.

In contrast to the pulse-direction interface to be described next, there is no corresponding limit switch status readout convention. If a CW or CCW limit switch is active, then the relevant control line's pulses should be inhibited by hardware from reaching the motor.

### *Pulse-direction memory-mapped interface*

An alternate form of motor interface specifies a bit pair as above, but the two bits are used as a pulse control and a direction control; i.e., the direction control line must be set first, and then the pulse is delivered. To specify this case, use aux byte values 0x10–0x17, assuming that the high word of the address is 0xFFFF. (Again, a variation for this interface allows using the value 0x10FF for the upper word of the full 32-bit I/O address. In this case, use aux values of 0x18–0x1F.) As in the CW-CCW case, four motors can be interfaced via one byte. In addition, there is a corresponding byte of status bit pairs that provide limit switch readouts. Two VME interface boards work this way, one made by Burr Brown and another by Ironics. The difference is that the Burr Brown board uses an address one byte higher for the corresponding status byte, whereas the Ironics board uses an address two bytes higher than that of the control byte. To specify these two cases, the low bit of the indicated bit# is odd for the Ironics case and even for the Burr Brown case. This same low bit of the indicated bit# also indicates the active state of the pulse control line. A zero low bit means the pulse is hi-active,

while a one low bit means the pulse is low-active.

In either the Burr Brown or the Ironics case, the indicated bit# is the pulse control line, and the alternate bit of the bit pair is the direction control line. The direction control bit state is 0 for CW and 1 for CCW. The limit switch state is 0 for off-limit and 1 for on-limit. The program logic will not issue pulses to a motor whose limit status reads 1. The CW limit corresponds to the indicated bit#, while the CCW limit is given by the alternate bit# of the status byte bit pair. Note that any motor interface that obeys these same conventions will work in the same way, even if neither Burr Brown nor Ironics manufactured it.

### *Any memory-mapped address*

The newest addition for motor support allows the use of an arbitrary memory-mapped address, rather than being limited to `0xFFFFxxxx` or `0x10FFxxxx`, where `xxxx` is the `ioAddr` word. This is specified by using `aux` values of `0xE $x$`  or `0xF $x$` . The `ioAddr` word specifies a digital Byte#, which means the `BADDR` Binary Address Table contains the full 32-bit address in the `BADDR` table entry for the given Byte#. In other respects, the support is the same as that for the two memory-mapped interfaces described above. The active pulse state is determined by the low-order bit of the full 32-bit address in this case, not the `ioAddr` word that is the Byte# index in the `BADDR` table. The specific motivation for this addition was to support motor control from an IRM digital I/O bytes, whose addresses are `0xFFF5810x`.

### *1553 pulse count interface*

For the case of a 1553-interfaced motor, the `aux` byte, concatenated with the `ioAddr` word, forms a 24-bit address of the 1553 command block that is to be used to send the 16-bit two's-complement step count to the motor controller. Current 1553 memory mapping therefore uses `aux` byte values of `0x2x`. The hardware delivers the pulses automatically. The system maintains a 150 Hz "shadow" counter, so that a user can know, by checking the motor count word (`listype#6`) for that channel, whether the motor controller has counted down. (Note that this is only valid if the motor controller operates at a 150 Hz pulse rate.)

### *SRM pulse count interface*

For motors run via a Smart Rack Monitor, the `aux` byte is used for the SRM arcnet node address, and it therefore uses values of `0xAx`. The `ioAddr` word in this case carries additional parameters to the SRM that will actually perform the motor step control. There are four such parameters, each occupying 4 bits. From the most significant nibble, they are as follows:

<i>Bits</i>	<i>Meaning</i>
15–12	Table#
11–8	Entry#
7–4	MotorType# 1=CW-CCW, 2=pulse-direction
3–0	Bit#

As an example, the GR2MID gradient control motor in Linac station #2 uses the value `0xB015` for its `ioAddr` word in its analog control field.

For the SRM case, the hardware does the stepping, so there is a "shadow" counter that is maintained by the system. As for the 1553 case, this counter can indicate whether the motor is still moving, if the hardware steps at 150 Hz.