PerformanceMotionDevices
+1 781 674 9860
engineering@pmdcorp.com

Release Notes
MC25x0 version 2.8

Document last updated: 2/5/2008

Product names: MC2540, MC2520, MC2510
Source control archive name: mc25x0 version 2.8
Production files: 25400028.bin, 25200028.bin, 25100028.bin
IO device: MC2000IOS2.0
Date of build: 1/30/2008

Description:
The MC2500 is a motion control processor for stepper motors and provides one to four axes of motion. This document details bug fixes and changes for this release.

Known Issues:

If the IO chip HostRdy signal (pin 8) is used for chip busy detection, the first instruction sent to the chipset after a power on or reset may be ignored or may produce a checksum error. It is recommended that in this configuration a NoOperation command be sent to the chipset as the first instruction after a power on or reset. If the *ReadStatus* operation is used to check the HostRdy state this problem does not occur.
To determine if the motion processor has generated an interrupt to the host, the host should check the state of the host interrupt signal. In most situations this signal should be connected to an external interrupt input on the host processor. The state of this signal is also reported by the *ReadStatus* operation but it is only accurate when the host is issuing actual chip commands and not simply polling *ReadStatus*.
When using the 1st generation compatibility mode and a counts to steps ratio of greater than one is being used, the values returned by GET_POS_ERR and GET_ACT_POS_ERR will be a modulus of the ratio.
When using the 1st generation compatibility mode and a count to step ratio of greater than one is being used, truncation in the value returned by GET_POS_ERR may occur when both the ratio is less than 16 and the position error limit is greater than 10,000. If a work around is necessary, the value sent in SET_POS_ERR can be increased to compensate for truncation.
When using the 1st generation compatibility mode and a counts to steps ratio of less than one is being used (very unusual), certain combinations of ratios and very large position error limits may cause truncation and/or may not be usable. Contact PMD for information.

Known Bugs:

A command sent via the serial interface that results in an instruction error does not set the instruction error bit in the event status register. The error code will be set in the status byte of the serial response packet.

Changes/Fixes:

Step Signal Output Changes

Corrected a problem in the AtRest signal which caused it to go active prior to the completion of a move.

Command Changes

Fixed - SrlEnable signal was intermittently staying active after an Update command.
Fixed a problem that resulted in serial communications locking up in the event that a command packet was sent by the host with additional (unnecessary) bytes.
Corrected a 1st generation command incompatibility. Previously the value of SET_POS_ERR was interpreted in units of steps. It is now interpreted in units of counts.
Corrected a 1st generation command incompatibility. Previously the value of GET_POS_ERR was returned in units of steps. It is now returned in units of counts.
Corrected a 1st generation command incompatibility. Previously the value of GET_ACT_POS_ERR was returned in units of steps. It is now returned in units of counts.
Corrected a 1st generation command incompatibility. Previously the value of bit 10 in the GET_MODE command (AUTO_UPDATE_MODE) had incorrectly polarity. Now the bit can be correctly interpreted as active low.

Profile Changes

Fixed a problem where a negative actual position was resulting in positive capture values when a non-unity encoder to step ratio was set.
Fixed a problem with S-Curve when certain move settings would cause a “Fractional Period”>1, which would cause the move to fail (excessive velocity and/or overshoot).
In trapezoidal profile mode, if the trajectory velocity is set to below the value of the StartVelocity once a trajectory is running, the trajectory should stop. Fixed.
In trapezoidal mode, if the trajectory velocity has been set to zero (SetVelocity 0) and the StartVelocity is > 0 issuing an update results in the trajectory jumping to -1. Fixed.
StopMode has no effect if StartVelocity is greater than zero. Fixed.

Miscellaneous Changes

Corrected a 1st generation encoder incompatibility issue. Since there is no 1st generation equivalent to the SetEncoderSource command, the occurrence of the SET_STEP_RATIO command will force the Encoder Source to Incremental.

Version 2.7

Changes/Fixes:

Command Changes

Resolved an issue listed in the known bugs for the previous version. Prior to this fix, if the Update instruction resulted in an instruction error the checksum returned by the chipset for the Update command would be incorrect.
Fixed a 1st generation command incompatibility. GET_ACTL_POS_ERR now behaves as expected. Previously this command returned CommandedPosition minus ActualPosition, Now it returns ActualPosition minus CommandedPosition.
The parameter range for SetTrackingWindow was increased from 0 to 2^{15}-1, to 0 to 2^{16}-1.

Profile Changes

Corrected a problem in velocity contouring profile mode related to very large velocity values. Prior to this fix, if the velocity was greater than.half of full-scale the velocity was interpreted as being negative instead of positive.
Corrected problems in s-curve profile mode when velocity or acceleration are set close to their maximum values.
Corrected a problem in s-curve profile mode related to move length. Previously, a move greater in length than half of full-scale position could only be done in the forward direction. Moves of greater than half of full-scale are now allowed in both directions.
Corrected problems in trapezoidal profile mode when velocity or acceleration are set close to their maximum values.
Resolved a problem in s-curve profile mode that resulted in a miscalculation during phase 5 of the motion. This only occurred for a specific combination of profile parameters. The error in calculation resulted in a position overshoot.

Step Signal Output Changes

none

Registers and Signals Changes

Corrected a problem in the activity status register related to the limit switches. Prior to this fix if both the positive and negative limit switches were active both switches would have to go inactive before either limit bit would be cleared. Now both bits are completely independent in operation.

Miscellaneous Changes

The minimum sample time per axis for this device has been increased from 102.4 microseconds to 153.6 microseconds.
If the auto stop mode is set to on, a motion error now results in the buffered velocity (SetVelocity) being set to zero. This prevents the condition where setting the motor mode to on if the axis is in velocity contouring profile mode after a motion error could cause motion to immediately restart.

Version 2.6

Known Issues:

If the IO chip HostRdy signal (pin 8) is used for chip busy detection, the first instruction sent to the chipset after a power on or reset may be ignored or may produce a checksum error. It is recommended that in this configuration a NoOperation command be sent to the chipset as the first instruction after a power on or reset. If the *ReadStatus* operation is used to check the HostRdy state this problem does not occur.
To determine if the motion processor has generated an interrupt to the host, the host should check the state of the host interrupt signal. In most situations this signal should be connected to an external interrupt input on the host processor. The state of this signal is also reported by the *ReadStatus* operation but it is only accurate when the host is issuing actual chip commands and not simply polling *ReadStatus*.

Known Bugs:

If the Update instruction results in an instruction error the checksum returned by the chipset for the Update command will be incorrect. This checksum error can be safely ignored. The GetHostIOError instruction should be issued to determine the cause of the instruction error. This bug will be corrected in the next release.
A command sent via the serial interface that results in an instruction error does not set the instruction error bit in the event status register. The error code will be set in the status byte of the serial response packet.

Changes/Fixes:

Command Changes

Fixed a 1st generation command incompatibility. SET_LMT_SENSE now behaves as expected. Previously the parameter for this command was inverted in comparison to its expected behavior.

Profile Changes

Corrected a problem in trapezoidal profile mode where a deceleration value of 1 could cause the profile to overshoot the destination position. Following the overshoot the profile would change direction and correctly complete at the destination position. This is now fixed.
Corrected a problem in trapezoidal profile mode where a non-zero value for the start velocity would prevent the profile from changing direction on the fly.
Corrected a problem where an axis in electronic gear profile mode could not be stopped using the SetStopMode command or a breakpoint with its action set to AbruptStop.

Step Signal Output Changes

Corrected a problem that could occur when the axis direction was changed on the fly. In this circumstance the pulse and direction signal could change state at the same time, resulting in a missed step. This is now fixed.

Registers and Signals Changes

The InMotion operation was changed to more accurately reflect the real state of motion. Now, after an Update command that will start motion the InMotion bit is immediately set. Previously the InMotion bit would not have been set for up to one chip cycle. This change corrected the situation where the MotionComplete bit would not be set for moves less than one step.

Miscellaneous Changes

none

Version 2.5

Known Issues:

If the IO chip HostRdy signal (pin 8) is used for chip busy detection, the first instruction sent to the chipset after a power on or reset may be ignored or may produce a checksum error. It is recommended that in this configuration a NoOperation command be sent to the chipset as the first instruction after a power on or reset. If the *ReadStatus* operation is used to check the HostRdy state this problem does not occur.
To determine if the motion processor has generated an interrupt to the host, the host should check the state of the host interrupt signal. In most situations this signal should be connected to an external interrupt input on the host processor. The state of this signal is also reported by the *ReadStatus* operation but it is only accurate when the host is issuing actual chip commands and not simply polling *ReadStatus*.

Known Bugs:

none

Changes/Fixes:

Command Changes

none

Profile Changes

Fixed a bug in velocity contouring mode that could allow a profile to restart after a motion error occurs when the SetMotorMode On command is issued. The profile will now only restart after a SetVelocity command has been issued followed by an Update.

Registers and Signals Changes

Resolved an issue in the SignalSense register where the StepOutput bit was inverted. This version now operates as documented generating a step on a high to low transition if the StepOutput bit is 0 and generating a step on a low to high transition if the StepOutput bit is 1.

Miscellaneous Changes

Resolved an issue in the limit switch handling where once in a limit an error could occur when further motion was attempted.
Resolved an issue where a motion error or SetMotorMode Off command would not stop the output of steps.

Version 2.4

Known Issues:

If the IO chip HostRdy signal (pin 8) is used for chip busy detection, the first instruction sent to the chipset after a power on or reset may be ignored or may produce a checksum error. It is recommended that in this configuration a NoOperation command be sent to the chipset as the first instruction after a power on

or reset. If the *ReadStatus* operation is used to check the HostRdy state this problem does not occur.
To determine if the motion processor has generated an interrupt to the host, the host should check the state of the host interrupt signal. In most situations this signal should be connected to an external interrupt input on the host processor. The state of this signal is also reported by the *ReadStatus* operation but it is only accurate when the host is issuing actual chip commands and not simply polling *ReadStatus*.

Known Bugs:

none

Changes/Fixes:

Command Changes

none

Profile Changes

Resolved a bug in the SCurve profile that resulted in an overshoot in phase 1 of the profile for extremely short moves. This would in turn create an error in the trajectory in Phase 5/6, which caused the trajectory to overshoot but settle at the correct destination.
Resolved a bug in the SCurve profile that would cause the trajectory to automatically restart if a ClearPositionError command was given after a move in the negative direction trajectory completed.
Resolved a bug in Velocity Contouring profile mode that could cause the motion complete bit to not be set if the axis was switched into VC on-the-fly from either SCurve or Trapezoidal profile mode.

Registers and Signals Changes

none

Miscellaneous Changes

Resolved an issue where the analog input values were incorrectly stored as signed values during trace. This would result in them being sign extended when they should not be because they are unsigned. They are now stored correctly.
Resolved an issue where parallel encoder input on one channel could corrupt the position of axes using incremental encoder input. This issue is now fixed.
Resolved an issue with analog inputs where the default conversion timing resulted in out of specification results. This has been fixed.

Version 2.3

Known Issues:

If the IO chip HostRdy signal (pin 8) is used for chip busy detection, the first instruction sent to the chipset after a power on or reset may be ignored or may produce a checksum error. It is recommended that in this configuration a NoOperation command be sent to the chipset as the first instruction after a power on or reset. If the *ReadStatus* operation is used to check the HostRdy state this

problem does not occur.

Known Bugs:

When SetInterruptMask is non-zero and the corresponding EventStatus bit is set, parallel host to I/O communications can become corrupted. The corruption causes the second and subsequent data reads to equal the first. If checksums are being read a checksum error will occur. If 32-bit values are being read the second word will equal the first which may result in unexpected data. If only 16-bit words are being read without checksum verification no errors will occur. This is the case with a normal interrupt service routine which sends GetInterruptAxis, GetEventStatus and ResetEventStatus.
If an Update or MultiUpdate is issued that would start the external profile motion AND a limit switch is currently active it will cause a chip reset. The workaround is to use any other profile mode to first move out of the limit before starting the external profile mode motion. If the chip enters a limit when the axis is already in external profile mode the chip behaves as expected, stopping motion if the LimitSwitchMode is set to enabled.

Changes/Fixes:

Command Changes

Resolved an issue that occured when a SetActualPosition or AdjustActualPosition resulted in a negative actual position. The problem was only "visible" if the ActualPositionUnits were set to “Steps” and the value set was not a multiple of the steps per rotation. The outcome was that the position returned by GetActualPosition would be –1 instead of the desired negative value.
Added MC1000 commands SET_POS_ERR and GET_POS_ERR

Profile Changes

none

Registers and Signals Changes

none

Miscellaneous Changes

none

Version 2.2

Known Bugs:

none

Changes/Fixes:

Command Changes

Resolved an issue that may have resulted in the bottom 4 bits of a SetSignalSense command having intermittent affect.

Profile Changes

Resolved an issue that prevented External Profile Mode from operating correctly with synchronous RAM.

Registers and Signals Changes

none

Miscellaneous Changes

none

Version 2.1

Known Bugs:

none

Changes/Fixes:

Command Changes

When using encoder feedback, there was a bug in Set/AdjustActualPosition where the number you set would be converted to and from the target units. This resulted in a rounding error where the number you set would not be the same value returned by a subsequent GetActualPosition. This is now fixed.
Fixed the condition where GetActualVelocity always returned the velocity in steps no matter what ActualPositionUnits was set to. Now ActualPositionUnits determines the units returned by GetActualVelocity.
When using encoder feedback, fixed a bug in SetMotorMode On and ClearPositionError that could cause the internally emulated step position to deviate from the actual step position when the EncoderToStepRatio was not equal to one.
Fixed a bug where GetStepLoopbackMode could not be called from the diagnostics port.
Fixed a bug where GetStartVelocity could not be called from the diagnostics port.

Profile Changes

Changes made to External Profile Mode as detailed below for compatibility with the Pathfinder contouring library.
External profile mode now responds correctly to limit switches. It also responds correctly to an AbruptStop set through a breakpoint or through a SetStopMode AbruptStop command. SmoothStop has no effect when the chipset is in external profile mode.
External profile mode now stops if it encounters an entry in the time buffer where the value is zero. This is a more convenient way of halting this profile mode than using a time or motion complete based breakpoint.

Registers and Signals Changes

none

Miscellaneous Changes

none

Version 2.0

Known Bugs:

none

Changes/Fixes:

Command Changes

none

Profile Changes

none

Registers and Signals Changes

none

Miscellaneous Changes

Version 2.0 and above of this chipset use a new IO that facilitates higher parallel communication speed. There is now no additional CP overhead if the checksum is read. “Get” commands can see a speed improvement of up to 30%.

In multi-drop serial mode, the chip previously did not respond with the expected status packet when a software “Reset” command was executed. This is now fixed.