Interfacing Microchip’s MCP41XXX and MCP42XXX Digital Potentiometers to Motorola’s 68HC12 Microcontroller

Author: Ezana Haile  
Microchip Technology, Inc.

OVERVIEW

The MCP41XXX and MCP42XXX family of digital potentiometers communicate using a standard 3-wire SPI™ compatible interface. This application note will cover communication between these devices and the Motorola 68HC12 family of microcontrollers. Specifically, the MC68HC912B32 evaluation board was used.

COMMUNICATION

Instructions for the MCP41XXX and MCP42XXX devices consist of two bytes. Figure 1 shows the format of these two bytes using a standard 3-wire SPI interface. The first byte is the command byte. The command byte contains which command to be issued and the instructions as to which potentiometer will execute the command (MCP42XXX devices contain two potentiometers). The second byte is the data byte. The MCP41XXX and MCP42XXX potentiometers are 8-bit or 256 tap potentiometers. All 8 bits in the data byte are wiper data bits.

There are four important bits in the command byte. Bits <4:5> determine which command is being issued and bits <0:1> determine which potentiometer will execute the command, see Figure 1. The MCP42XXX device contains two pots, P0 and P1. Bits <0:1> of the command byte allow the user to select either, both, or neither potentiometer. A ‘1’ for either P1 or P0 will cause the data to be written to the respective data register and a ‘0’ for P1 or P0 will cause no change. The MCP41XXX device contains only one potentiometer. For this device, P1 is a don’t care.

Bits <4:5> determine which command is being issued. For the MCP41XXX and MCP42XXX devices, there are three possible commands:
1. Write new data to potentiometer(s).
2. Shutdown potentiometer(s).
3. NOP (No Operation).

![Diagram of SPI interface and command byte format]

FIGURE 1: INSTRUCTION SEQUENCE AND COMMAND BYTE SUMMARY FOR DIGITAL POTENTIOMETERS.

SPI™ is a trademark of Motorola Inc.
IMPLEMENTATION

Appendix A has assembly code for the 68HC12 family of microcontrollers using hardware SPI implementation. First, this code initializes the necessary registers to enable the hardware SPI. See Set register values for SPI in the source code in Appendix A. Once the registers are set to the proper configuration, chip select is cleared to enable the digital potentiometer. Accumulator A is then loaded with the command byte to be transmitted. The content of accumulator A is transferred to the SPI register, SP0DR. A branch to the Transmit subroutine transmits the command byte to the digital potentiometer through the SPI port. Once transmission is complete, the data byte is loaded into accumulator A followed by a transfer to the SPI register, SP0DR. Another branch to the Transmit subroutine sends the data to the digital potentiometer. Once data transmission is complete, driving the chip select line high sets the wiper position to the corresponding value. The software interrupt command, SWI, completes the digital potentiometer programming sequence.

Note: The SS/CS output can not be used as a chip select line for the MCP41XXX/MCP42XXX. This integrated feature of the SPI module is driven high after transmitting a byte which prematurely terminates the programming sequence. Therefore, an I/O line must be dedicated to drive chip select high at the end of the second transmitted byte.

CONCLUSION

This application note shows how to interface Microchip’s MCP41XXX/MCP42XXX Digital Potentiometers to the Motorola 68HC12 family of microcontrollers. The source code in Appendix A is compatible with the 68HC11 family of microcontrollers. The MC68HC912B2 evaluation board was used for this application.

REFERENCES

Software License Agreement

The software supplied herewith by Microchip Technology Incorporated (the “Company”) is intended and supplied to you, the Company’s customer, for use solely and exclusively on Microchip products.

The software is owned by the Company and/or its supplier, and is protected under applicable copyright laws. All rights are reserved. Any use in violation of the foregoing restrictions may subject the user to criminal sanctions under applicable laws, as well as to civil liability for the breach of the terms and conditions of this license.

THIS SOFTWARE IS PROVIDED IN AN “AS IS” CONDITION. NO WARRANTIES, WHETHER EXPRESS, IMPLIED OR STATUTORY, INCLUDING, BUT NOT LIMITED TO, IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE APPLY TO THIS SOFTWARE. THE COMPANY SHALL NOT, IN ANY CIRCUMSTANCES, BE LIABLE FOR SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES, FOR ANY REASON WHATSOEVER.

APPENDIX A: ASSEMBLY CODE USING HARDWARE SPI IMPLEMENTATION

*****************************************************************************************
*                                                                                       *
*        Interfacing Microchip’s MCP42xxx digital potentiometer to                      *
*        Motorola's 68HC12 microcontroller using Serial Peripheral Interface (SPI)      *
*                                                                                       *
*****************************************************************************************

*                                                                                       *
*         Filename:      MOT_6812_2_MCP_Digipot.asm                                     *
*         Date:          01.09.2001                                                     *
*         File Version:  1.00                                                           *
*                                                                                       *
*         Controller:    MC68HC912B32                                                   *
*                                                                                       *
*         Assembler:     WIN IDE V1.13   P&E Microcomputer Systems, Inc.                *
*         Programmer:    Prog12Z programmer V1.09                                       *
*         Debugger:      ICD12Z debugger V1.24                                          *
*                                                                                       *
*         Author:        Ezana Haile                                                    *
*         Company:       Microchip Technology, Inc.                                     *
*                                                                                       *
*****************************************************************************************

*                                                                                       *
*         Microchip's MCP42xxx Digital Potentiometer (POT) requires serial               *
*         communication to program the command and data bytes.  This program             *
*         demonstrates how Motorola's 68HC12 microcontroller is interfaced to the POT    *
*         using SPI.  The assembly code for the Motorola microcontrollers are fully       *
*         compatible, therefore, this code can be used in other controllers such as       *
*         68HC11.  However, register addresses must be modified accordingly.              *
*                                                                                       *
*         To change the command byte and the POT wiper position the user must change     *
*         COMMAND and DATA variables properly and reprogram the controller.              *
*                                                                                       *
*****************************************************************************************

* ---------------------------------------------
*     Equate registers addresses
* ---------------------------------------------

copct1 equ $0016 ;COP Control Register
mode equ $000b ;mode register
reset equ $ffe ;reset
sp0cr1 equ $00d0 ;spi control register 1
sp0cr2 equ $00d1 ;spi control register 2
sp0br equ $00d2 ;spi baud rate register
sp0sr equ $00d3 ;spi status register
sp0dr equ $00d5 ;spi data register
ports equ $00d6 ;port s data register
ddrs equ $00d7 ;data direction register for port s
purses equ $00db ;pull-up and reduced drive for port s
cs equ $0004 ;chip select (ps2)
*-------------------------------------------------
*     Digital POT command and data bytes
*-------------------------------------------------
command   equ     $13             ;command byte for the digital pot
data       equ     $50             ;digital value to set the wiper position

*-------------------------------------------------
*     Set register values for SPI
*-------------------------------------------------
org     $8000           ;program the Flash EEPROM

start    ldaa    #$19            ;special single chip mode
          staa    mode
          ldaa    #$00            ;turn off watchdog
          staa    copctl
          ldaa    #$5e
          staa    sp0cr1          ;set control register 1
          clra
          staa    sp0cr2          ;set control register 2
          staa    sp0br           ;set baud rate
          staa    purds           ;set pull-up and reduced drive for ports
          ldaa    #$ff
          staa    ddrs            ; set data direction register for ports

*-------------------------------------------------
*     Program the POT
*-------------------------------------------------
bclr    ports,cs        ;select digital pot
          ldaa    #command        ;load the command byte
          staa    sp0dr           ;store command byte to the SPI data register
          bsr     transmit        ;program the command byte
          ldaa    #data           ;load the data byte
          staa    sp0dr
          bsr     transmit        ;program the data byte
          bset    ports,cs        ;deselect digital pot
          swi                     ;software interrupt

*-------------------------------------------------
*     Transmit data thru SPI
*-------------------------------------------------
transmit  brset   sp0sr,$80,done  ;wait until the end of transmission
          bra     transmit

done      ldaa    sp0sr
          ;clear SPIF (SPI interrupt request)
          ldaa    sp0dr
          ;return from subroutine

*-------------------------------------------------
*     Set reset vector
*-------------------------------------------------
org     reset           ;set reset vector
(dw     start)          ;upon reset goto 'start'

*-------------------------------------------------
Information contained in this publication regarding device applications and the like is intended through suggestion only and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. No representation or warranty is given and no liability is assumed by Microchip Technology Incorporated with respect to the accuracy or use of such information, or infringement of patents or other intellectual property rights arising from such use or otherwise. Use of Microchip’s products as critical components in life support systems is not authorized except with express written approval by Microchip. No licenses are conveyed, implicitly or otherwise, under any intellectual property rights.

Trademarks

The Microchip name and logo, the Microchip logo, PIC, PICmicro, PICMASTER, PICSTART, PRO MATE, KeeLoq, SEEVAL, MPLAB and The Embedded Control Solutions Company are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

Total Endurance, ICSP, In-Circuit Serial Programming, Filter-Lab, MXDEV, microID, FlexROM, fuzzyLAB, MPASM, MPLINK, MPLIB, PICC, PICDEM, PICDEM.net, ICEPIC, Migratable Memory, FanSense, ECONOMONITOR, Select Mode and microPort are trademarks of Microchip Technology Incorporated in the U.S.A.

Serialized Quick Term Programming (SQTP) is a service mark of Microchip Technology Incorporated in the U.S.A.

All other trademarks mentioned herein are property of their respective companies.

© 2001, Microchip Technology Incorporated, Printed in the U.S.A., All Rights Reserved.

Printed on recycled paper.