

# Motorola Semiconductor Application Note

---

## AN1757

## Add a Unique Silicon Serial Number to the HC05

By **Mark Glenewinkel**  
Field Applications Engineering  
Consumer Systems Group  
Austin, Texas

### Introduction

---

Many embedded systems require serial numbers to help track printed circuit boards, identify nodes on a network, register product, and provide security access.

This application note describes the interface between an HC05 microcontroller (MCU) and the DS2401 silicon serial number from Dallas Semiconductor. The DS2401 provides a factory-lasered, 64-bit ROM number that is unique to each device. The address bus structure uses a 1-Wire<sup>TM</sup> interface that reduces the overhead of control, data, address, and power pins. One pin on the DS2401 combines all of these functions.

The 1-Wire interface can also be used with the DS2502, a device similar to the DS2401. In addition to the 64-bit ROM number, it has 1024 bits of user-programmable EPROM. The memory residing in the DS2502 allows the user to program the device to hold information about a node or a system. Typical applications include storage of calibration constants, security access codes, and system revision status.

---

1-Wire is a trademark of Dallas Semiconductor.



Circuitry and example code are included to demonstrate the interface between the DS2401 and the HC05. Although no example code is given for the DS2502, the serial drivers designed for the 1-Wire bus make an easy API (application programming interface) for the user to add application-specific functions to utilize the DS2502's additional features.

### DS2401/DS2502 Features

---

#### DS2401 Features

The DS2401 provides these features:

- Unique, factory-lasered and tested 64-bit registration number ensures that no two parts are alike.
- Multiple DS2401s can be identified on the bus
- Reduces control, address, data, and power to a single data pin
- Directly connects to a single port pin of an MCU
- Communicates up to 16.3 Kbits per second
- Zero standby power required
- Low-cost TO-92, SOT-223, and 6-pin TSOC packages
- 1-Wire bus communicates over a wide voltage range of 2.8 volts to 6.0 volts from  $-40\text{ }^{\circ}\text{C}$  to  $+85\text{ }^{\circ}\text{C}$

#### DS2502 Features

In addition to those features on the DS2401, the DS2502 provides these features:

- 1024 bits of user programmable EPROM
- EPROM partitioned into four 256-bit pages for randomly accessing packetized data
- Each memory page can be permanently write-protected to prevent unwanted tampering of the device.
- Device allows the user to program additional EPROM bits without disturbing the existing data.

## Description

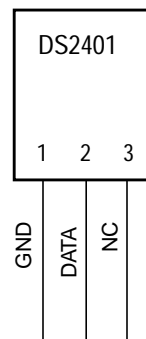
---

The DS2401 has a 64-bit lasered ROM that is unique to each device. The first eight bits signify the 1-Wire Family member being addressed. The next 48 bits are unique to each DS2401 device, allowing more than 281 trillion different devices to be in the field. The last eight bits are a CRC (cyclic redundancy check) of the first 56 bits. The 64-bit ROM is retrieved by using the 1-Wire bus. The 1-Wire bus allows multiple DS2401s to be on the bus. The 1-Wire bus network controller circuitry has a search algorithm embedded to determine the identity of each device on the bus.

## DS2401 Hardware Interface

---

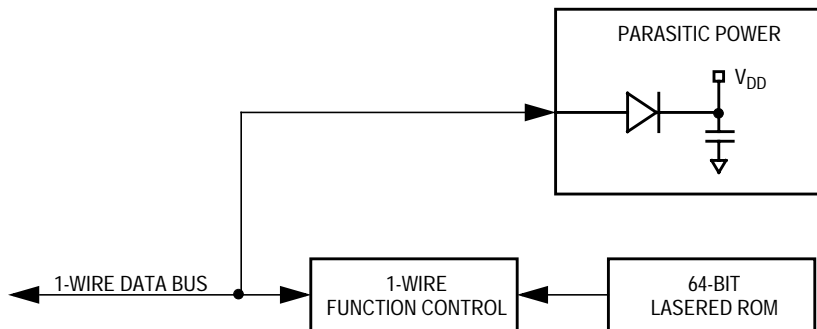
### Pinout and Pin Descriptions



**Figure 1. DS2401 TO-92 Pinout**

The bidirectional DATA pin is the only interface pin to a microcontroller. Parasitic power is derived from the required pullup resistor on the DATA pin. No other power input is needed for the DS2401. All data transceived between the master and the DS2401 is read and written least significant bit (LSB) first.

## Block Diagram



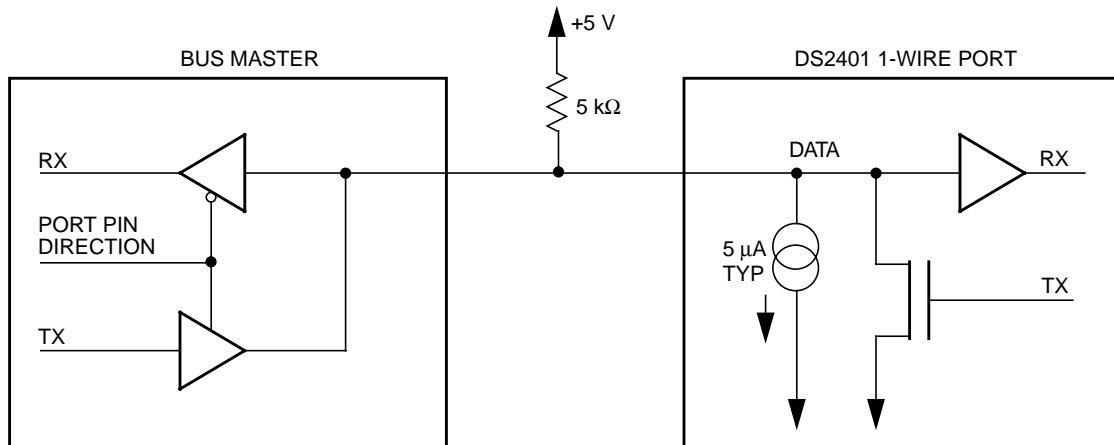
**Figure 2. DS2401 Block Diagram**

## 1-Wire Interface

**Figure 3** shows the hardware interface of the 1-Wire bus. The bus has a single master and one or more slave devices. In all cases, the DS2401 is a slave.

It is important that each device on the bus be able to drive it at the appropriate time. Thus, each device must have open drain or three-state outputs. The maximum bus rate allowed is 16.3 kbits per second.

The idle state of the bus is high. If for any reason a transaction is suspended, the bus must be left in the idle state if the transaction is to resume at a later time. If the bus is held low for more than 120  $\mu$ s, one or more of the slave devices could be reset. A pullup resistor is required on the bus to ensure proper idling of the bus and to provide parasitic power to the DS2401.



**Figure 3. 1-Wire Bus Interface**

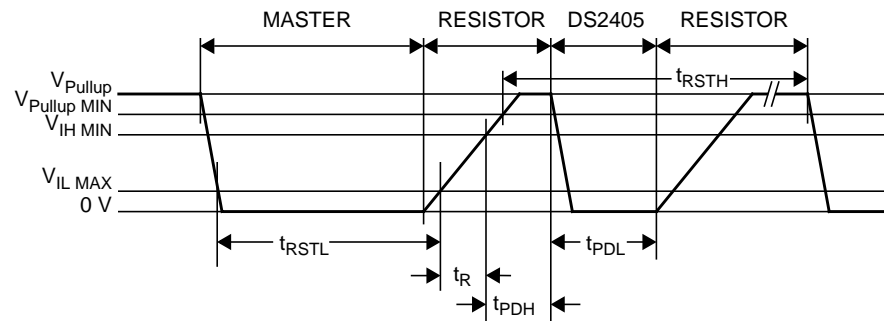
### 1-Wire Timing

The 1-Wire protocol is divided into two types of transactions. These are:

- Reset and presence pulse
- Write and read one bit of data

When a device is idling in the high state, the master starts communicating to the DS2401 by issuing a reset pulse. The master must drive the bus low for at least 480  $\mu\text{s}$ . After this time, the master turns its port pin into a high impedance input pin and allows the pullup resistor to bring the bus back high. Over the next 480  $\mu\text{s}$ , the master reads the bus looking for a low. If the DS2401 is active and ready to communicate, it will drive the bus low. If the master does not receive a presence pulse, further communication cannot occur.

**Figure 4** shows the reset and presence pulse timing.



$480 \mu\text{s} \leq t_{\text{RSTL}} < 960 \mu\text{s}$   
 $480 \mu\text{s} \leq t_{\text{RSTH}} < \text{INFINITE}$ , INCLUDES RECOVERY TIME  
 $15 \mu\text{s} \leq t_{\text{PDH}} < 60 \mu\text{s}$   
 $60 \mu\text{s} \leq t_{\text{PDL}} < 240 \mu\text{s}$

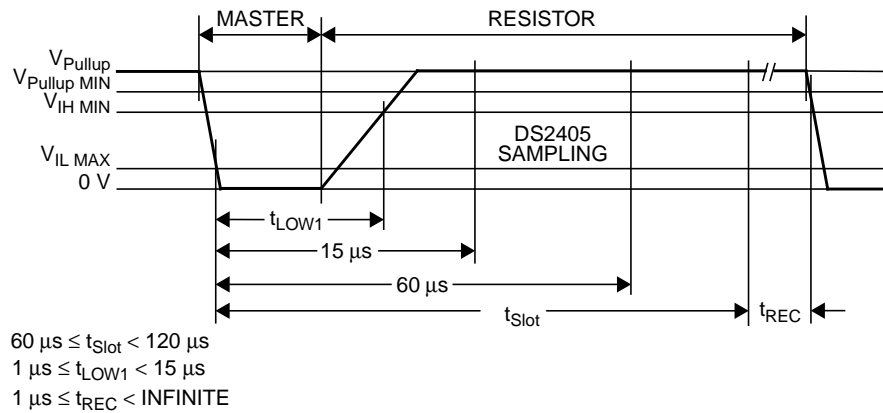
**Figure 4. Reset and Presence Pulse Timing**

After the presence pulse is received, data now may be communicated between the master and the slave. A bit is transceived by specific time slots that are initiated by the master sending a falling edge sync pulse. The sync pulse defines the start of a time slot that is at least 60  $\mu\text{s}$  long. After this time slot is finished, a recovery time of at least 1  $\mu\text{s}$  is required to give the DS2401 time to respond to the next bit being transmitted. The time slot and recovery time together add up to 61  $\mu\text{s}$ , which defines the maximum communication speed of 16.3 kbits per second.

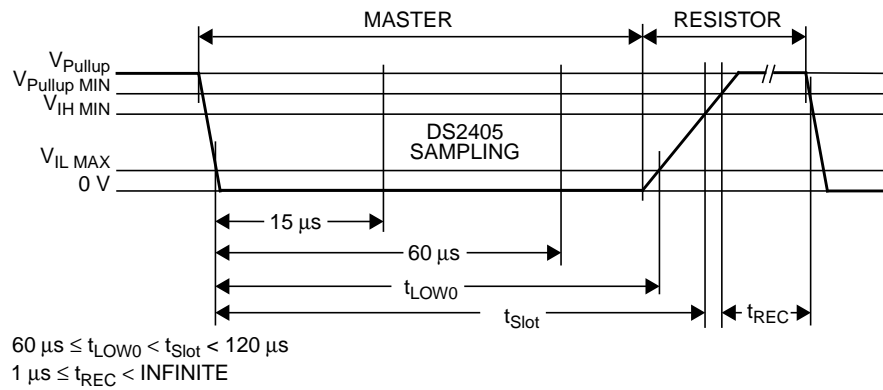
Three different time slots can be generated. They are:

- Write-one time slot
- Write-zero time slot
- Read data time slot

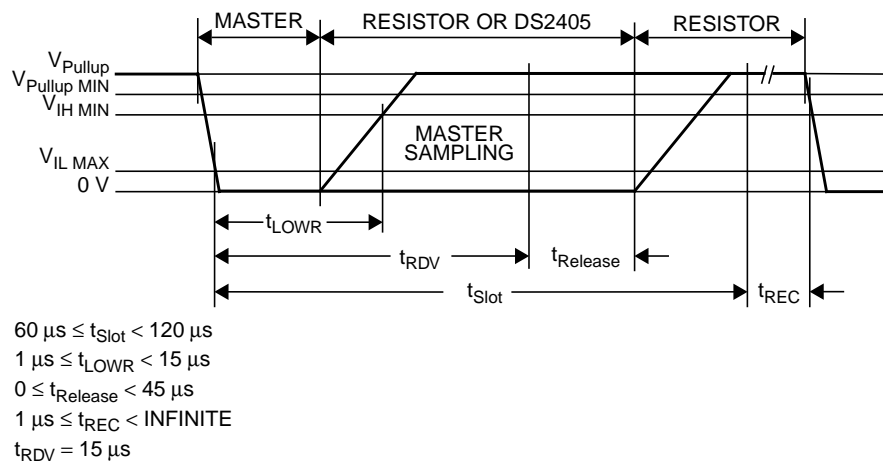
The timing diagrams for these time slots are shown in [Figure 5](#), [Figure 6](#), and [Figure 7](#).



**Figure 5. Write-One Time Slot**



**Figure 6. Write-Zero Time Slot**



**Figure 7. Read Data Time Slot**

A step-by-step example of the protocol needed to read the DS2401's 64-bit ROM code is:

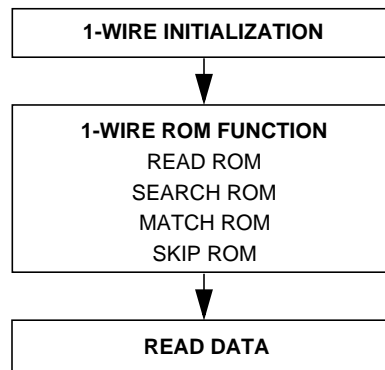
1. The master transmits a reset pulse.
2. The master waits for the presence pulse from the DS2401. Once detected, go to step 3.
3. The master sends out the read ROM command to the DS2401. The code for read ROM is \$33 or %00110011 and is sent out LSB first.
  - a. Write-one — 2 times
  - b. Write-zero — 2 times
  - c. Write-one — 2 times
  - d. Write-zero — 2 times
4. After the read ROM function has been sent, the DS2401 is ready to transmit its 64-bit code contents on the bus. The master sends out 64 read data time slots to read each bit off the bus and store them in its memory.
5. The transaction is now complete. To issue another command, the DS2401 must be reset again.



## DS2401 Software Interface

---

The transaction sequence to access the DS2401 over the 1-Wire bus is illustrated in **Figure 8**.



**Figure 8. DS2401 Transaction Sequence**

### Initialization

All transactions start with a device initialization routine. This is accomplished by the master sending a reset pulse and then reading a presence pulse from the DS2401.

### ROM Function Commands

Once the bus master has detected a presence pulse, one of the ROM commands can be issued. All ROM functions are eight bits long.

#### *Read ROM — \$33*

This command allows the bus master to read the DS2401's unique 64-bit ROM code. This command can be used only if there is a single DS2401 on the bus. Otherwise, a data collision will occur. All 1-Wire devices have this 64-bit lasered ROM. The first eight bits signify the 1-Wire Family. The next 48 bits are unique to the DS2401. The last eight bits are a CRC (cyclic redundancy check) of the first 56 bits. Consult the DS2401 data sheet for more detail on the CRC.

#### *Search ROM — \$F0*

When a system has many devices on the bus, the master may not know the number of devices on the bus or their 64-bit ROM codes. This command is useful when reading the 64-bit ROM code of all devices on the bus. The search ROM command uses a tedious process of

elimination to determine the identity of the devices on the bus. A three-step process is used on each bit of the 64-bit code.

The steps are:

1. The master reads a bit position of the 64-bit code.
2. The master reads the complement of the bit position.
3. With this information, the master writes a bit on the bus to match those devices that have the same bit value. This in turn deselects the other devices that do not have a matched bit.

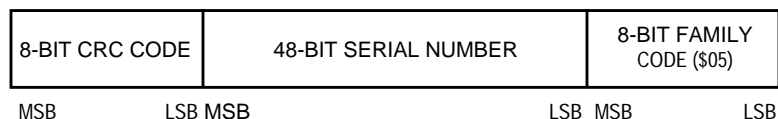
This process is repeated until all devices eventually are eliminated except the device that matches exactly to the 64-bit code that was just transmitted.

*Match ROM — \$55*  
*Skip ROM — CC*

The complete protocol of the 1-Wire bus includes a command to bypass the 64-bit ROM code and a command to match the 64-bit ROM code. Since the DS2401 contains only the 64-bit ROM with no additional data fields or functions, these commands are not applicable to the DS2401.

### Memory Map

The DS2401 has no memory other than the 64-bit ROM code. The ROM code is shown in [Figure 9](#).



**Figure 9. DS2401 Memory Map**

## MC68HC705C8A Hardware Interface

---

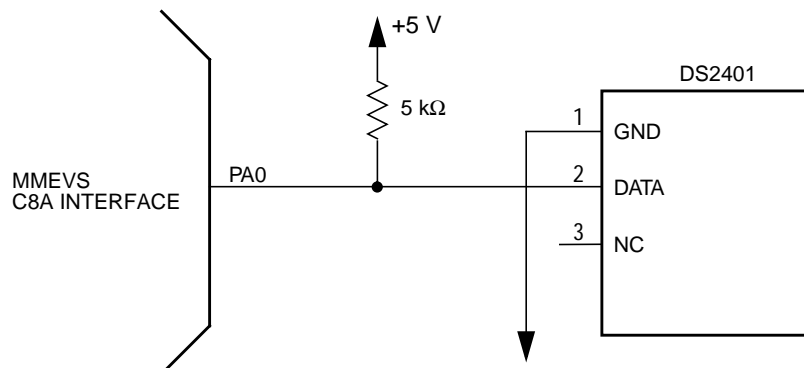
This application note uses the MC68HC705C8A (C8A) member of the HC05 Family to test the interface between the DS2401 and the HC05. The C8A is one of the most popular members of the HC05 Family.

The MC68HC705C8A has:

- 7744 bytes of erasable programmable read-only memory (EPROM)
- 176 bytes of RAM
- 24 I/O (input/output) pins
- Seven input-only pins
- Peripherals
  - Serial peripheral bus (SPI)
  - Serial communications interface (SCI)
  - 16-bit capture/compare timer

The schematic used for testing the C8A to DS2401 interface on the MMEVS development system is shown in [Figure 10](#). Bit 0 of port A is used to transmit and receive data on the DATA pin of the DS2401.

For further information on the HC705C8A, consult the *MC68HC705C8A Technical Data*.



**Figure 10. C8A to DS2401 Interface Test Circuit**

### MC68HC705C8A Software Interface

---

Input/output driving or manipulation is the process of toggling I/O pins with software instructions to create a certain hardware peripheral. The HC05 CPU provides special instructions to specifically manipulate single I/O pins.

The serial transmission driver has been put into three subroutines:

- **RESET\_PULSE** — Sends out a reset pulse to the DS2401 and waits for a presence pulse. If no presence pulse is found, the routine goes into an error loop.
- **TXD** — Takes a byte of data and creates eight time slots of either write-one or write-zero, depending on the bit being transmitted.
- **RXD** — Transmits eight read data time slots. Each bit is read and shifted into a byte of RAM.

The flowcharts for the DS2401 serial I/O drivers are shown in [Figure 11](#), [Figure 12](#), and [Figure 13](#).

The flowchart for the main test routine is in [Figure 14](#). The step-by-step sequence of tests is:

1. Read ROM. This command asks the DS2401 to put its 64-bit ROM code on the bus.
2. Set up a loop to retrieve a total of eight bytes.
3. Store each byte into the RAM buffer DS2401\_ROM. The location DS2401\_ROM+7 will have the contents of the LSB of the ROM code. The location DS2401\_ROM will have the contents of the MSB of the ROM code.

This routine demonstrates the interface software needed to communicate with the DS2401. Although the C8A was used, any HC05 device could utilize this interface code. Minor adjustments of port pins and memory maps might be necessary.

The assembly code for the test routine is provided in [Code Listing](#).

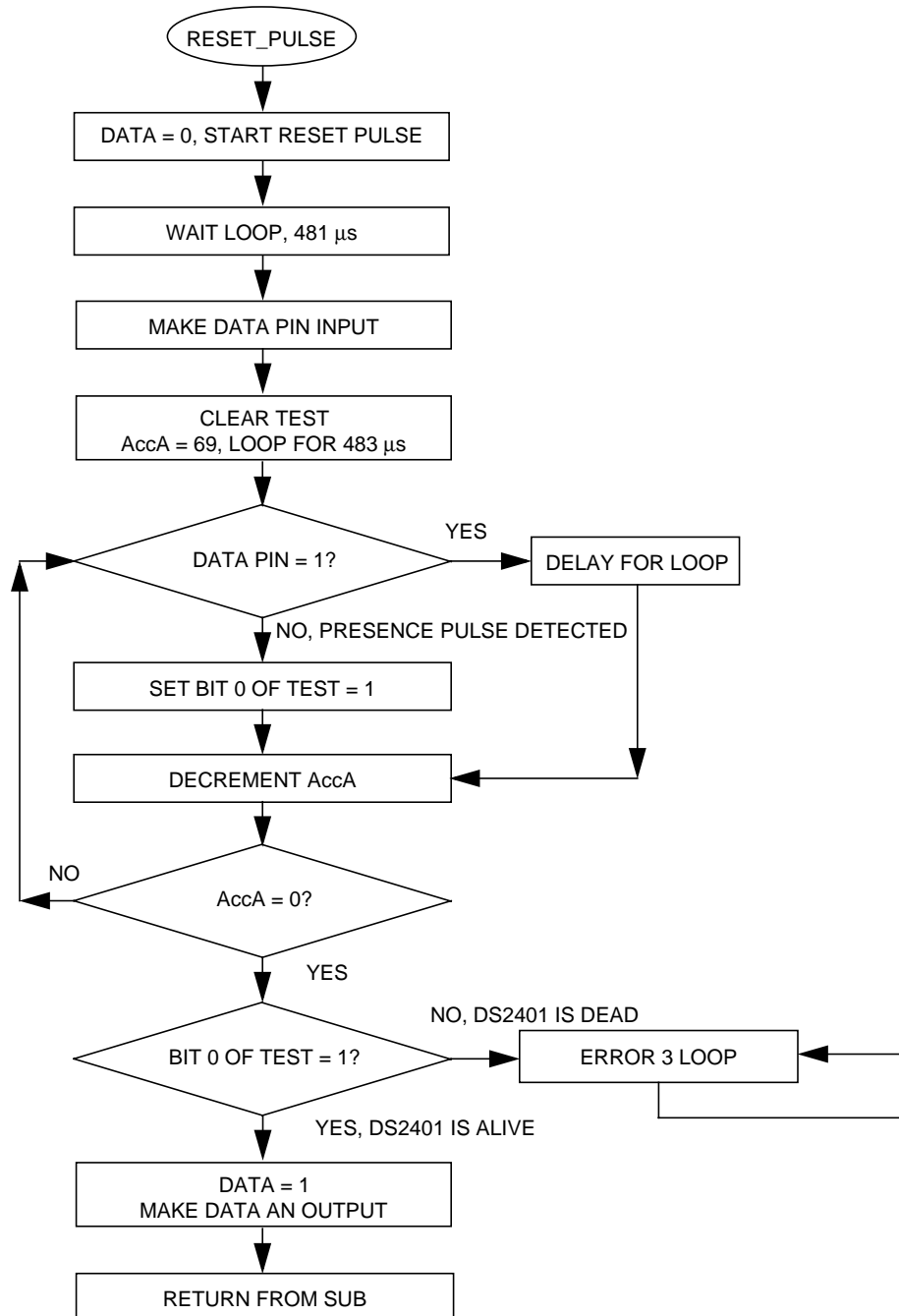
## Development Tools

---

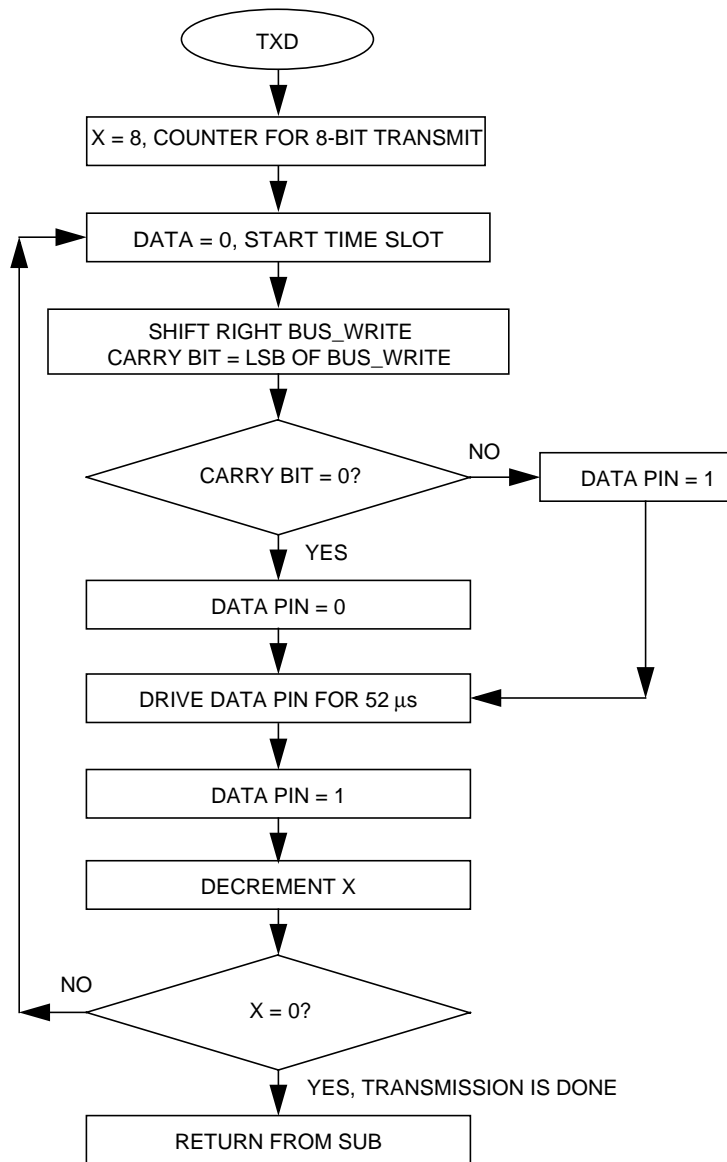
The interface was created and tested using these development tools:

- M68MMPFB0508 — Motorola MMEVS platform board
- M68EM05C8A — Motorola C series emulation module
- Win IDE Version 1.02 — Editor, assembler, and debugger by P&E Microcomputer Systems

## Flowcharts for the Serial Drivers



**Figure 11. RESET\_PULSE Subroutine Flowchart**



**Figure 12. TXD Subroutine Flowchart**

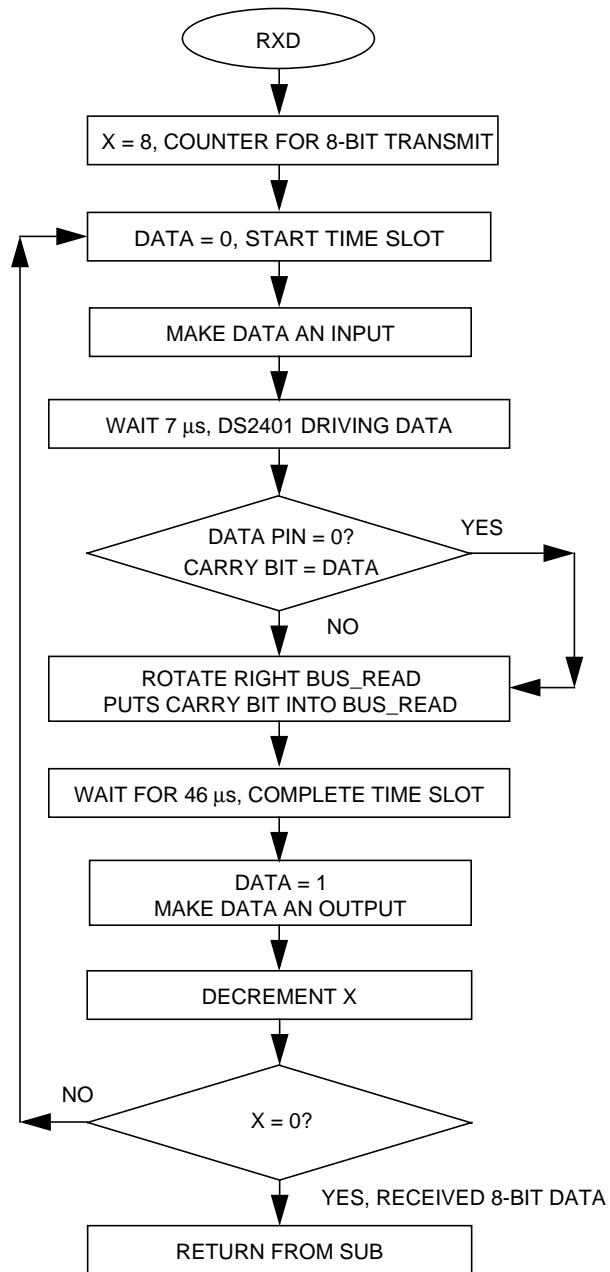


Figure 13. RXD Subroutine Flowchart



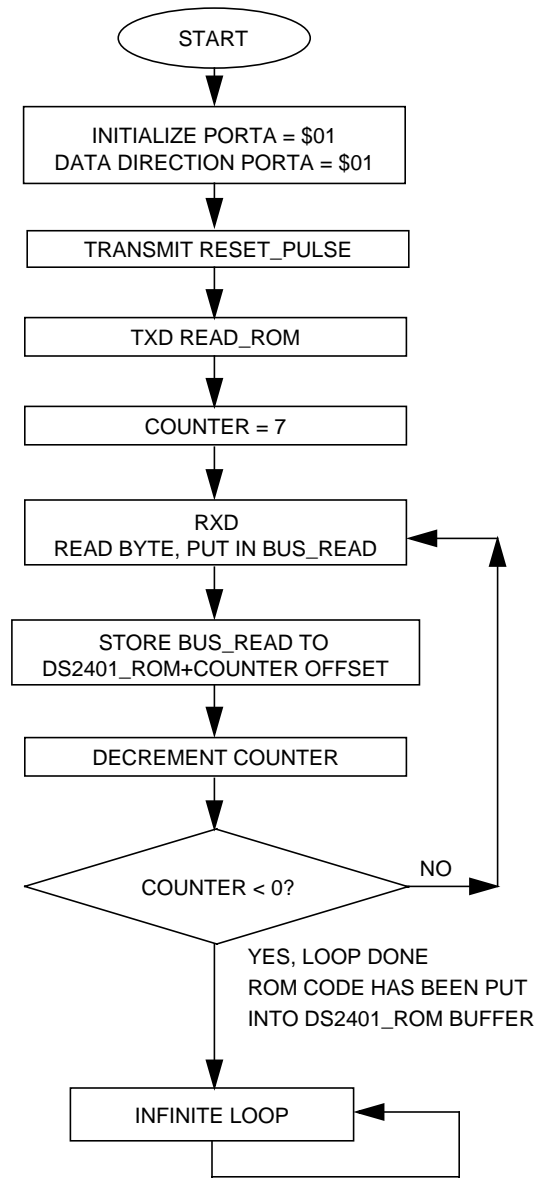


Figure 14. Flowchart for Main Test Routine

## Code Listing

---

```
*****
*
* File name: DS2401.ASM
* Example Code for the MC68HC705C8A Interface to the
*   Dallas DS2401 Silicon Serial Number
* Ver: 1.0
* Date: June 19, 1998
* Author: Mark Glenewinkel
*   Motorola Field Applications
*   Consumer Systems Group
* Assembler: P&E IDE ver 1.02
*
* For code explanation and flow charts,
* please consult Motorola Application Note
*   "Add a Unique Silicon Serial Number to the HC05"
*   Literature # AN1757/D
*
* NOTE: All timing functions are based on a 2MHz internal bus clock
*
*****

*** SYSTEM DEFINITIONS AND EQUATES *****
*** Internal Register Definitions
PORTA      EQU      $00          ;PortA
DDRA       EQU      $04          ;data direction for PortA

*** Application Specific Definitions
DATA       EQU      0T          ;PortA, bit 0, data signal
DATA_DIR   EQU      0T          ;PortA Data Dir for DATA signal

*** ROM Function Commands
READ_ROM   EQU      $33          ;read ROM
MATCH_ROM  EQU      $55          ;match ROM
SEARCH_ROM EQU      $F0          ;search ROM
SKIP_ROM   EQU      $CC          ;skip ROM

*** Memory Definitions
EPROM      EQU      $160         ;start of EPROM mem
RAM         EQU      $50         ;start of RAM mem
RESET      EQU      $1FFE        ;vector for reset

*** RAM VARIABLES *****
          ORG      RAM
BUS_WRITE  DB      $00          ;storage for byte write on bus
BUS_READ   DB      $00          ;storage for byte read on bus
TEST       DB      $00          ;test result for presence
DS2401_ROM RMB      8           ;storage for DS2401 64-bit ROM code
```

```

;DS2401_ROM = MSB
;DS2401_ROM+7 = LSB
;temporary counter
COUNTER      DB          $00

*** MAIN ROUTINE *****
                ORG          EPROM          ;start at begining of EPROM
*** Intialize Ports
START         lda          #$01          ;init PORTA
                sta          PORTA
                lda          #$01          ;config outputs on PORTA
                sta          DDRA

*** Issue "READ ROM" command
                jsr          RESET_PULSE   ;send reset on bus
                lda          #READ_ROM
                sta          BUS_WRITE
                jsr          TXD           ;send Read ROM cmd

                lda          #7T
                sta          COUNTER       ;set counter to rxd 8 bytes

ROM_Cycle     jsr          RXD           ;receive data from bus - 1 byte
                lda          BUS_READ
                ldx          COUNTER
                sta          DS2401_ROM,x ;store byte read into DS2401_ROM
                dec          COUNTER
                bpl          ROM_Cycle     ;loop done?

DUMMY        bra          DUMMY          ;test sequence is over

*** SUBROUTINES *****
*** Routine creates a reset pulse and then checks for the
*** presence pulse from the DS2401
*** If no presence pulse, goto error loop

* Create a greater then 480usec reset pulse
RESET_PULSE  bclr          DATA,PORTA
                lda          #160T          ;2 wait for 481usec
J1           deca
                bne          J1           ;3
                bclr          DATA,DDRA   ;DATA is now an input

* Wait for greater than 480usec, look for presence pulse,
* TEST will be equal to $01 if presence is detected
                clr          TEST
                lda          #69T          ;wait for 483usec
J2           brset          DATA,PORTA,J3 ;5 DATA=1?
                bset          0,TEST      ;5 DATA=0, presence detected

```

## Application Note

```

        bra          J4                ;3   set TEST bit 0 = 1

J3      brn          J3                ;3   branch has same time
        brn          J3                ;3
        nop          ;2

J4      deca        ;3   decrement Acca
        bne          J2                ;3   done?

* Check TEST, if TEST=$01, then ok
* if TEST=$00, then goto error routine
        brset       0,TEST,J5          ;TEST bit 0 = 1?
ERROR3  bra          ERROR3            ;presence pulse not detected, error
J5      bset        DATA,PORTA       ;TEST passed, DATA=1
        bset        DATA,DDRA       ;DATA is output now
        rts

*** Routine takes contents of BUS_WRITE and transmits it serially to
*** it serially on the bus, LSB first
TXD     ldx         #8T                ;set counter

* Drive DATA=0 for 9usec
WRITE   bclr        DATA,PORTA       ;5 DATA=0, start time slot

        asr         BUS_WRITE         ;5 Carry bit = LSB
        bcc         J6                ;3
        bset        DATA,PORTA       ;5 DQ=1
        bra         J7                ;3 branch to clock_it
J6      bclr        DATA,PORTA       ;5DQ=0
        brn         J6                ;3evens it out

* At this point, 10.5usec has expired and either a 1 or 0
* is being transmitted on the DATA pin
* We must now wait for at least 49.5usec, routine below is 52usec
J7      lda         #17T               ;2
J8      deca        ;3
        bne         J8                ;3

* Make sure DATA=1, then wait for more than lusec for recovery time
        bset        DATA,PORTA       ;5 DATA=1
        decx        ;3
        bne         WRITE             ;3 all 8 bits transmitted?
        rts          ;return from sub
```

```

*** Routine clocks the bus to read data from DATA, LSB first
*** 8 bit contents are put in BUS_READ
RXD          ldx          #8T          ;set counter

* Drive DATA=0 for 1µsec,
READ         bclr        DATA,PORTA   ;5 DATA=0, start time slot
            nop          ;2
            bclr        DATA,DDRA     ;5 make DATA an input

* Wait for 7 usec, then sample DATA
            lda          #2T          ;2
J9           decr        ;3
            bne         J9           ;3

* Now read data and wait for 50µsec
            brclr       DATA,PORTA,J10 ;5 carry bit = DATA
J10          ror         BUS_READ      ;5 carry bit into BUS_READ

* At this point, 15.5µsec has expired since time slot started
* We must now wait for at least 44.5µsec, routine below is 46µsec
            lda          #15T         ;2
J11          decr        ;3
            bne         J11          ;3

* Make sure DATA=1, then wait for more than 1usec for recovery time
            bset        DATA,PORTA   ;5 DATA=1
            bset        DATA,DDRA     ;5 make DATA an output
            decx        ;3
            bne         READ          ;3 all 8 bits received?
            rts          ;return from sub

*** VECTOR TABLE *****
            ORG          RESET
            DW           START

```

## References

---

*MC68HC705C8A Technical Data*, Motorola document order number MC68HC705C8A/D, 1996.

*M68HC05 Applications Guide*, Motorola document order number M68HC05AG/AD, 1996.

*DS2401 Datasheet*, Dallas Semiconductor, 1997.

*DS2502 Datasheet*, Dallas Semiconductor, 1997.

HC05/08 Website


<http://design-net.com/csic/welcome.htm>

Development Tools Website

<http://design-net.com/csic/devsys/sg173/sg173.htm>



# Application Note

Motorola reserves the right to make changes without further notice to any products herein. Motorola makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Motorola assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters which may be provided in Motorola data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including "Typicals" must be validated for each customer application by customer's technical experts. Motorola does not convey any license under its patent rights nor the rights of others. Motorola products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Motorola product could create a situation where personal injury or death may occur. Should Buyer purchase or use Motorola products for any such unintended or unauthorized application, Buyer shall indemnify and hold Motorola and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Motorola was negligent regarding the design or manufacture of the part. Motorola and  are registered trademarks of Motorola, Inc. Motorola, Inc. is an Equal Opportunity/Affirmative Action Employer.

## How to reach us:

**USA/EUROPE/Locations Not Listed:** Motorola Literature Distribution, P.O. Box 5405, Denver, Colorado 80217, 1-800-441-2447 or 1-303-675-2140. Customer Focus Center, 1-800-521-6274

**JAPAN:** Nippon Motorola Ltd.: SPD, Strategic Planning Office, 141, 4-32-1 Nishi-Gotanda, Shinagawa-ku, Tokyo, Japan. 03-5487-8488

**ASIA/PACIFIC:** Motorola Semiconductors H.K. Ltd., 8B Tai Ping Industrial Park, 51 Ting Kok Road, Tai Po, N.T., Hong Kong. 852-26629298

**Mfax™, Motorola Fax Back System:** RMFAX0@email.sps.mot.com; <http://sps.motorola.com/mfax/>;

TOUCHTONE, 1-602-244-6609; US and Canada ONLY, 1-800-774-1848

**HOME PAGE:** <http://motorola.com/sps/>

Mfax is a trademark of Motorola, Inc.



**MOTOROLA**

© Motorola, Inc., 1998

AN1757/D