

Motorola Semiconductor Application Note

AN-HK-32

In-Circuit Programming of FLASH Memory in the MC68HC908GP32

By T.C. Lun
Applications Engineering
Microcontroller Division
Hong Kong

This application note describes In-Circuit Programming (ICP) of the FLASH memory in the Motorola MC68HC908GP32 (GP32) microcontroller, a general purpose device based on the HC08 architecture that has 32k-bytes of on-chip FLASH.

The text is divided into two parts:

- PART 1 — covers a general overview of ICP and techniques that can be applied to the GP32
- PART 2 — covers a simple low-cost ICP implementation on the GP32

For detailed specification on MC68HC908GP32, please refer to the datasheet: Motorola order number MC68HC908GP32/H.

PART 1 Introduction

In-circuit programming is a process by which the device is programmed or erased with the device on the final circuit board — the *target system*. This allows the *user code* to be changed without having to remove the device off the target system for reprogramming or initial programming.

On GP32, the 32k-bytes FLASH memory is allocated for the user code, with an additional 36-bytes of FLASH for user defined reset and interrupt vectors. A high voltage supply is not required by the GP32 for program or erase operations; as it is generated by an internal charge-pump. This

FLASH memory can be programmed or erased using software routines running either in *User mode* or *Monitor mode*, by writing to the FLASH Control register at address \$FE08.

User Mode	In User mode, the GP32 is running the user code, that has been programmed in the FLASH memory. This is the mode in which the GP32 will be running during most of the time.
Monitor Mode	In Monitor mode, the GP32 is running code that has been permanently programmed into an area of memory in the GP32 during fabrication. The monitor code is used for communicating to an external host, connected via a serial link. Programming an initially blank GP32 FLASH memory is executed in monitor mode.
Initial FLASH Programming	The mode in which the GP32 enters is latched after a power-on-reset (POR), and depends on the logic level on the following pins: \overline{IRQ} , \overline{RST} , PTA0, PTA7, PTC0, PTC1, and PTC3. (For details, please refer to the Monitor ROM section in the datasheet.)

In-Circuit Programming in User Mode

ICP in user mode can be implemented so as to maintain target system operation while reprogramming the FLASH memory in the GP32. Reprogramming the FLASH memory in the GP32 involves two stages. The first stage is an erase operation to erase the existing data in the FLASH memory cell. The minimum erase size is 128-bytes, known as a *page*. The MASS bit in the FLASH Control register provides the option for erasing the entire FLASH array in one operation, known as *MASS erase*. It should be noted that an erase byte of FLASH memory reads as \$FF. The second stage is the programming process, which programs the blank FLASH memory with new data. Thus, reprogramming involves: erase and program.

ICP code	Performing ICP in user mode requires that the erase and programming routines — the <i>ICP code</i> — are to be stored in a part of non-volatile memory that can be called by the user program. This means the ICP code needs to be a routine that is part of the user code, and programmed into GP32's FLASH memory. With this in mind, ICP in user mode cannot be performed if the FLASH memory is initially blank; a blank device. Initial blank devices are programmed in Monitor mode (see next section for ICP in Monitor Mode).
----------	---

With the ICP code programmed into the FLASH memory, it is called by software or hardware, and can operate in two ways:

The ICP code sets up the GP32 a communication link with an outside host system via the GP32 port pins or the SCI interface, and then transfers control of the GP32 MCU to the host system. The host issues commands to erase the GP32's FLASH memory and downloads data to program the FLASH memory. In this case, the GP32 ICP code is acting as a command interpreter.

Alternatively, the ICP code can carry out the erase process and downloads new data from an external source for the programming. The source can be an intelligent host or an EPROM containing the new user code.

In both of the above methods, the ICP code must be loaded into the RAM area of memory, and the routine executed in the RAM area. Program or erase operations are not allowed while program is running in the FLASH area. If it was possible for the ICP code to execute in the FLASH area, there is the danger of erasing the ICP code itself.

Block Protected FLASH Memory

There is one situation where the FLASH memory cannot be erased in user mode: when it is *block protected*. The FLASH Block Protect register at address \$FF7E is used to protect (prevent from erase or programming) a block of, or the entire FLASH memory. Once that block of memory is protected, it cannot be erased while the GP32 is running in user mode. It can only be erased in monitor mode. The FLASH Block Protect register cannot be rewritten to "unblock" the protected FLASH, because it is implemented as FLASH register and is also block protected.

More details on ICP in user mode using the SCI for host communication is discussed in the Motorola Application Note *AN1770: In-Circuit Programming of FLASH Memory in the MC68HC908GP20*.

In-Circuit Programming in Monitor Mode

In Monitor mode, the GP32 is running the *monitor code* that has been permanently programmed into an area of memory (\$FE20 to \$FF52) in the GP32 during fabrication. First time programming of the GP32's FLASH memory can only be executed in monitor mode.

The monitor code consists of routines for communicating to a host connected using a serial link via pin PTA0. Once the link is established, control of the MCU is transferred to the host system. The host controls the MCU by directly writing to the MCU registers. Monitor mode can be entered in two ways:

High Volt Entry to Monitor Mode Similar to most Motorola MCUs, providing a high voltage ($V_{DD}+2.5V$ for GP32's case) on the \overline{IRQ} pin during a POR will force the GP32 to enter monitor mode. With this high voltage entry method, the clock input to the MCU (at OSC1) must be either 4.9152MHz or 9.8304MHz. This clock divides to produce the 9600 baud communication speed on PTA0; since the GP32's PLL is disabled in monitor mode (see Monitor mode section in datasheet for more details).

Blank Vector Entry to Monitor Mode With the new FLASH memory implementation, there was a need to reduce the number of wire connections to the target system to program the MCU when ICP was required. The other method for entry to monitor mode is a blank reset vector. The only time when a reset vector is blank is when the entire GP32's FLASH memory is blank — the reset vector can only be erased during a mass erase operation. If the \overline{IRQ} pin was also grounded during the mode latch after POR, the PLL is enabled, allowing the use of a 32.768kHz crystal between OSC1 and OSC2 for the input clock. This is a saving of two wires and three jumpers compared with the high voltage entry method: no connection to \overline{IRQ} and OSC1.

Implementing ICP in monitor mode has the advantage that no ICP code needs to be written for the user code. In addition, the *MCUscribe* program, a free Motorola utility, is available for the PC host system that talks to the MCU via PTA0 serial link.

Other ICP Considerations

Signal Conditioning Normal system activities will usually be halted during an ICP operation, to allow an uninterrupted programming process. Therefore, at the start of the ICP process, the MCU should be configured such that no pin contention or runaway signal will occur during the ICP process. Also note that when the system is first switched-on with a MCU having a blank FLASH memory, the port pins default to their reset states.

Pin Isolation If the MCU pins used for connecting to the external host are shared with the target system, make sure they are isolated to the proper logic level when the ICP connection is made.

PART 2

Introduction

The following ICP method is low-cost; with minimal system and user code changes. It involves two steps:

1. Erasing the FLASH memory in User mode.
2. Programming the FLASH memory in Monitor mode (blank vector entry) using Motorola's SPGMR08 Serial Programmer.

Constraints

This ICP method must meet these two conditions:

1. The FLASH memory is not block protected.
Block protected FLASH memory cannot be erased in user mode. A high voltage (V_{TST}) applied to the \overline{IRQ} pin is required for entering monitor mode to erase the FLASH memory.
2. The bus frequency must be at 2.4576MHz for programming the FLASH (see Programming the FLASH Memory in Monitor Mode). The 2.4576MHz is used to derive the 9600 baudrate for the communication between MCU and Host. The 2.4576MHz bus frequency can be generated using the MCU's PLL (usually with a 32.768kHz external reference crystal). If the PLL is not used, then the external crystal must be 9.8304MHz (4 times the bus frequency).

Mass Erasing the FLASH Memory in User Mode

The program listing at the back of this application note contains the routine for mass erasing the MCU. Since this program is for demonstration purposes, only the MASS_ERASE subroutine is required for inclusion to the user program. Other parts of the program involves setting up of the bus clock using the PLL and the polling of the pin PTA0 and PTA7 for calling the ICP routine.

What the program does is this:

1. Configure the PLL for 2.4576MHz bus from 32.768kHz.
2. Check logic levels on PTA0 and PTA7; if true, proceed to mass erase.
3. Load MASS_ERASE routine to RAM memory.
4. Execute MASS_ERASE routine. The routine loops until the reset vector is blank.

In this implementation, PA0 and PA7 are used for setting up a request for mass erase operation. After a POR, when PA0 = 1 and PA7 = 0 (see figure 1), the user code will load the mass erase routine into RAM and perform a FLASH mass erase operation.

In the erase routine, the delay timing is based on a bus frequency of 2.4576MHz, and the mass erase operation is repeated until the user vectors and the security bytes are erased. The time required for the mass erase operation is less than two seconds.

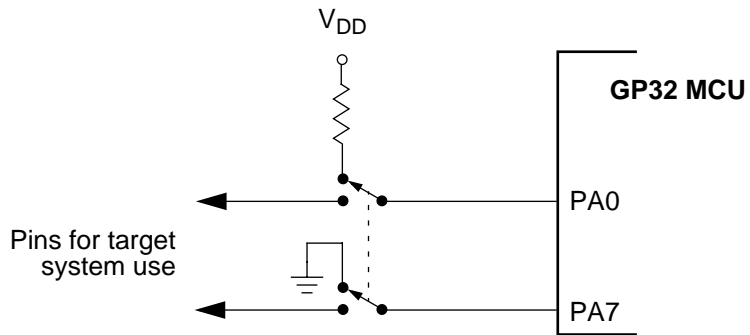


Figure 1. Mass Erase Port Pin Configuration

The flowchart in figure 2 shows the sequence of events for the mass erase operation.

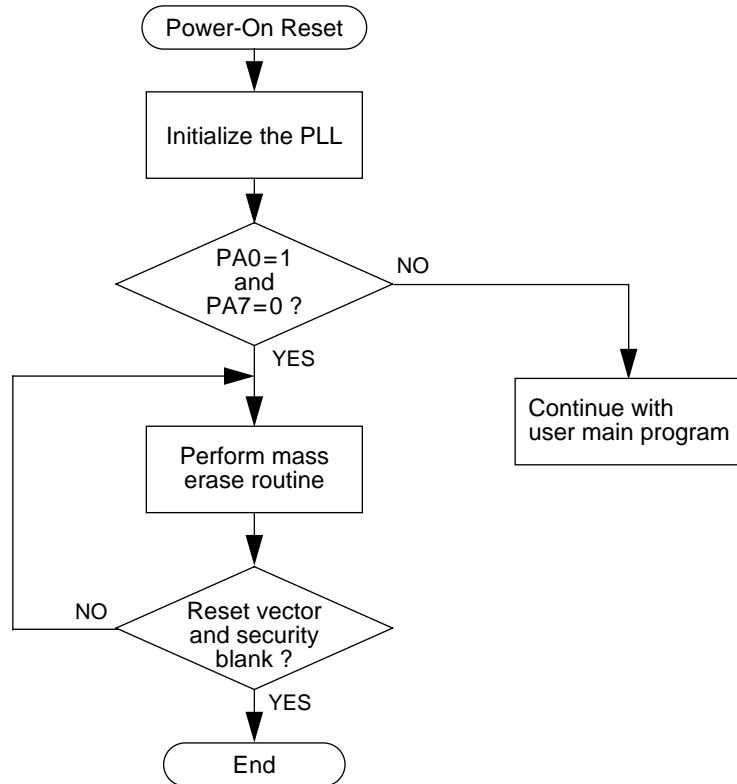


Figure 2. Mass Erase Flowchart

Procedure for mass erase

Using the sample program, this step-by-step procedure erases the GP32 FLASH in user mode:

1. Switch off the power to the target system.
2. Isolate port pins PA0 and PA7 from target system logic.
3. Set PA0 to high via a pull-up resistor to V_{DD} .
4. Set PA7 to ground directly to V_{SS} .
5. Switch on the power to the target system.
6. Wait 2 seconds.
7. Switch off power to the target system.
8. FLASH memory is now erased.

The next section describes the procedure for programming the GP32 FLASH memory using blank vector entry to monitor mode.

Programming the FLASH Memory in Monitor Mode

Programming the GP32's blank FLASH memory is achieved by running the MCU in monitor mode; and with a host connected using a serial link. Monitor mode can be entered in one of two ways after a power-on-reset:

- A high voltage ($V_{DD}+2.5V$) applied the \overline{IRQ} pin, or
- The FLASH memory is erased blank.

The latter method for entering monitor mode for programming the FLASH memory will be described here. With this method, the MCU enters monitor mode after a power-on reset when it detects that the reset vector, \$FFFE–\$FFFF, is blank (containing \$FF).

The Motorola *SPGMR08 serial programmer* is used as the interface between the target system and the PC host system.

Figure 3 shows the connection to the SPGMR08. Three wires are used:

- PA0 — This is the serial data link between the host and the MCU.
- V_{DD_S} — This line provides power and power-on reset synchronization between the host and MCU.
- GND — Common ground for the systems.

The other two wires are only necessary for a high voltage entry to monitor mode.

For this implementation, PA7 and \overline{IRQ} are required to be grounded for the mode entry, and the crystal frequency must be 32.768kHz.

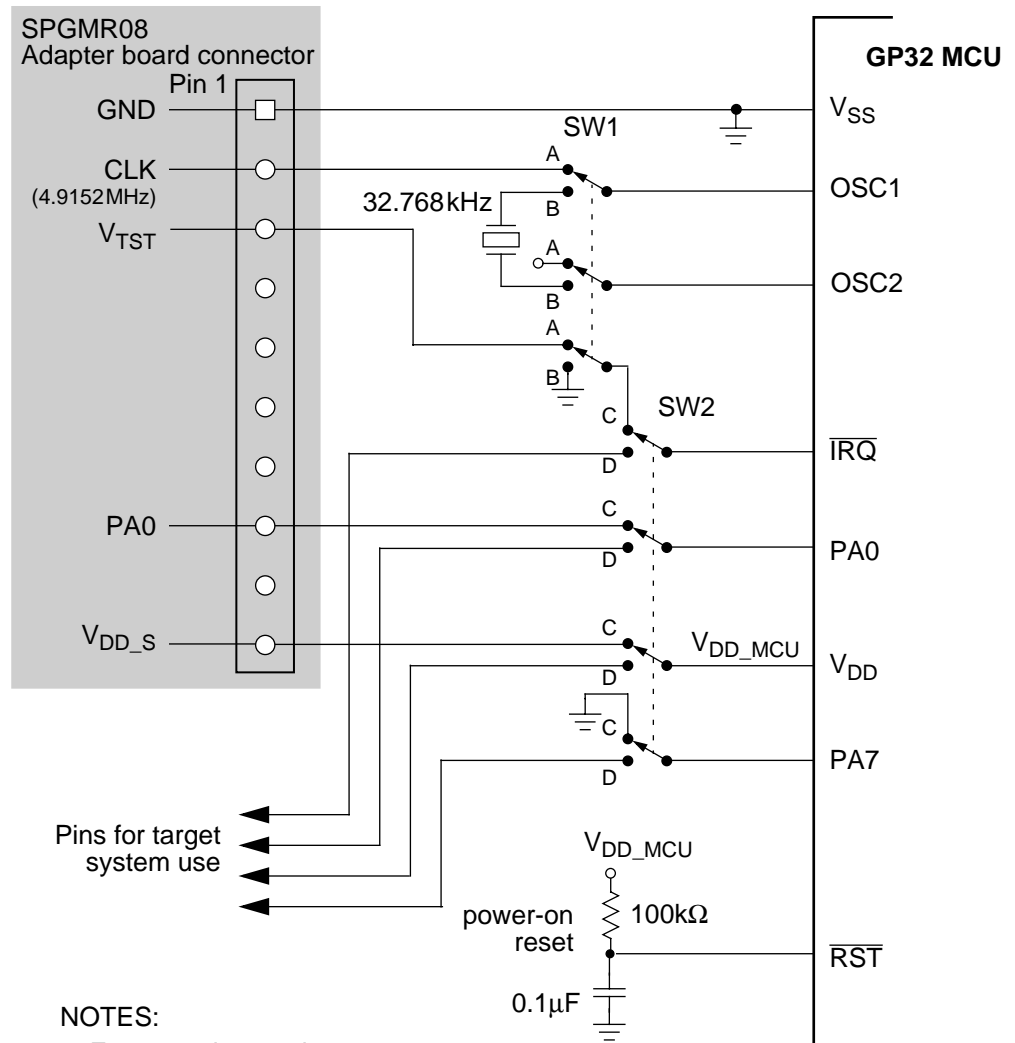


Figure 3. Programming Setup

Once the programming system is connected as in figure 3, the programming is carried out by running the *MCUscribe* utility supplied with the SPGMR08. When *MCUscribe* has finished programming, set the jumpers back to their original position, and then select the "power-off" command on the *MCUscribe* utility screen menu.

Further Information

The above ICP method has two limitations. They are:

1. The erase and program operations are for the entire 32k-bytes of FLASH memory — An erase operation erases all FLASH locations; a program operation programs all FLASH locations.
2. There must be no power outage during erase or program operations; otherwise, a high voltage must be applied to the $\overline{\text{TRQ}}$ pin so that the MCU can enter Monitor mode. The alternative is to extract the MCU off the target system and reprogrammed using an external programmer.

Further cost-savings can be achieved by using the circuit in figure 4 to replace the SPGMR08.

Other ICP methods that can be applied to the GP32 are described in the application note: *AN1770 — In-Circuit Programming of FLASH memory in the MC68HC908GP20*.

Program Listing

```
;-
; Assembler Directives
; $base          10t
;-
; 68HC908GP32 User Mode FLASH Mass Erase
; Author       : T.C. Lun
; File Name    : gp32icp.asm
;-
; Description
; This program allows the MCU to mass erase itself in user mode.
; The detect condition for mass erase is PA0=1 & PA7=0.
;
; Jumper setting:
;           Jumper          Mass Erase
;           -----          -
;           PA0             Pullup(10K)
;           PA7             Short to GND
;
; Memory usage:
;
; RAM        $00A0-$00DF
; FLASH      $F000-$F06F
;-
; Version      Date          Description
; 1.1          12/28/99      Changed to PA7 from PA3 for erase jumper
* PLL EQUATES
PCTL          equ  $0036      ; PLL Control Register
PBWC          equ  $0037      ; PLL Bandwidth Control Register
PMSH          equ  $0038      ; PLL Multiplier Select Register High
PMSL          equ  $0039      ; PLL Multiplier Select Register Low
PMRS          equ  $003A      ; PLL VCO Range Select Register
PMDS          equ  $003B      ; PLL Reference Divider Select Register
AUTO          equ  7          ; Bit 7 of PBWC
LOCK          equ  6          ; Bit 6 of PBWC
PLLON         equ  5          ; Bit 6 of PCTL
BCS           equ  4          ; Bit 4 of PCTL
* Initial Settings for 32.768 kHz crystal clock to produce a 2.4576 MHz
* internal clock
P             equ  0          ; PLL Prescaler Program Bits (PRE)
; value of PCTL (def = 0)
E             equ  1          ; PLL VCO Power-of-Two Range Select Bits
; (VCR) value of PCTL (def = 0)
NHI          equ  1          ; PLL Multiplier Select Bits (MUL)
; value of PMSH (def = 0)
NLO          equ  $2C        ; PLL Multiplier Select Bits (MUL)
; value of PMSL (def = 0)
L            equ  $80        ; PLL VCO Range Select Bits (VRS)
; value of PMRS (def = 64)
R            equ  0          ; PLL Reference Divider Select Bits
; (RDS) value of PMDS (def=1)
; 0 value for R or N is interpreted as a 1
```

```

;-----
; Flash Control Register
;-----
FLCR          equ    $fe08 ; Flash Control Register
HVEN          equ    3
MASS          equ    2
ERASE         equ    1
PGM           equ    0
;
FLBPR         equ    $ff7e ; Flash Block Protect Register
;
CONFIG1       equ    $1F
CONFIG2       equ    $1E
;
PTA           equ    $00
DDRA          equ    $04
;
RAM           equ    $40
;
MAIN          equ    $F000
FLPROGST      equ    $FF19
USERVECTOR    equ    $FFFE
VECTOR        equ    $FFF0
;-----
; Main Program
;-----
                org    MAIN
START:
    rsp
    clr        CONFIG2
    mov        #$31, CONFIG1    ; Disable COP & LVI
    clr        PCTL             ; Set Bus frequency = 2.4576MHz
    bset       0,PCTL           ; using the external 32.768KHz
    bset       0,PMSH          ; crystal and turn on PLL
    lda        #NLO
    sta        PMSL
    lda        #L
    sta        PMRS
    bset       PLLON, PCTL
    bset       AUTO, PBWC
    brclr     LOCK, PBWC,*
    bset       BCS, PCTL
    clr        DDRA             ; check user mode mass erase condition
                                ; PA0=5V & PA7=GND in user mode condition
    brclr     0,PTA, USERCODE ; check PA0 = 5V
    brset     7,PTA, USERCODE ; check PA7 = GND
    clrh
    clrx
NEXTRAM:
    lda        MASS_ERASE, x    ; Load mass erase code from flash to RAM
    sta        RAM, x
    incx
    cbeqcx    #ENDRAM-MASS_ERASE, RUNRAM
    bra       NEXTRAM

```

```

RUNRAM:      jmp      RAM          ; Execute the mass erase

USERCODE:   bra      *          ; Start of the user code
;-----
; Mass Erase
;-----

MASS_ERASE:
    ldhx     #FLCR          ; set HX = FLCR
    lda      #%00000110
    sta      ,x             ; Set MASS & ERASE flag
    lda      FLBPR         ; read from block protect register
    sta      FLBPR         ; write BPR in security fail
    lda      #4            ; delay 10us
    bsr      DELAY

    lda      #%00001110
    sta      ,x             ; set HVEN
    lda      #40
    bsr      DLY_N100US    ; delay 4ms

    lda      #%00001100
    sta      ,x             ; clear ERASE flag
    bsr      DLY_100US     ; delay 100us
    clra
    sta      ,x             ; clear HVEN flag
    clrh
    ldx      #$05

V_CHECK:
    incx
    lda      VECTOR,x      ; load security and user vectors
    cmp      #$FF          ; check blank
    beq     V_NEXT
    jmp     RAM            ; bluk erase again if vectors are
                        ; not blank

V_NEXT:
    cpx     #$0F           ; check last location
    bne     V_CHECK

ICPMODE:
    bra     *             ; Waiting for power-off

;-----
; Delay Subroutine
; - for bus frequency = 2.4576MHz
;-----
; 5 + 4 + N*(1+3) = 9+4N

DLY_100US:                                     ; [4] : 100.5us
    lda     #59                                     ; [2]


DELAY:
    deca                                       ; [1]
    bne     DELAY                               ; [3]
    rts                                         ; [5]

```

```
DLY_N100US:                ; [4]
    psha                    ; [2]
    bsr      DLY_100US      ; [247]
    pula                    ; [2]
    deca                    ; [1]
    bne      DLY_N100US    ; [3]
    rts                    ; [5]
```

ENDRAM:

```
    org      USERVECTOR
    fdb      START                ; RESET
```

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

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

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, Inc., 2000



MOTOROLA
