Semiconductor Products Sector
Application Note

AN2150

Converting the HC08AZ32 ROM Code for the HC08AZ32A

by Tracy McHenry
Systems Engineering
East Kilbride, Scotland.

1 Introduction

The purpose of this document is to help customers to convert 0.65µ HC08AZ32 ROM codes for 0.5µ HC08AZ32A. It highlights the differences between the two devices and provides a step by step guide to converting HC08AZ32 ROM codes for the HC08AZ32A. This application note is also applicable to ROM code conversions from the HC08AB32 to the HC08AB32A, the HC08AB24 to the HC08AB24A, the HC08AZ16 to the HC08AZ16A and the HC08AZ24 to the HC08AZ24A. However, it is *not* applicable to ROM code conversions from the HC08AB16 to HC08AB16A.

The main change is in the EEPROM module. The EEPROM is made from a new NVM technology. Read operations remain unchanged, however, program and erase operations are a super-set of the present algorithm. The EEPROM contains 2 new registers that must be set up correctly before any attempt is made to program or erase the EEPROM. The new registers are required to provide the EEPROM with a constant timebase of 35µs from the user's oscillator frequency. The HC08AZ32A EEPROM also supports a new option that enables significantly faster programming/erasing of the EEPROM. Appendix A shows the recommended algorithms for the program and erase operations.

It is very important to spend time gaining familiarity with the EEPROM changes as it is essential that the EEPROM module is set up correctly before any program or erase operations are called. Failure to do so could cause premature wear out of the EEPROM or could result in improper programming/erasing of the EEPROM.

Also, care should be taken over the bit settings of the MOR registers. The polarity of several bits in the MORA register has been changed. Extra bits are available in MORB which also has a new address, $FE09. The MOR register changes have been made to align the HC08AZ32A to the other HC(9)08 A-family devices.

2 Major differences between the HC08AZ32 and the HC08AZ32A

This section describes the differences between the HC08AZ32 and the HC08AZ32A. Each affected module is listed along with a summary of the changes.

2.1 Mask Option Register A

The polarity of several bits in this register has changed to align the HC08AZ32A with other HC08AS & AZ family devices.

Bit-2, COPRS – COP Rate Select bit to determine the timeout period of the COP – selected as a ‘1’ now enables a short COP timeout period of 8176 cycles.

Bit-4, LVIPWR – LVI Power enable bit (previously LVIPWRD) – selected as a ‘1’ now enables the LVI module power.

Bit-5, LVIRST – LVI reset enable bit (previously LVIRSTD) selected as a ‘1’ enables the reset signal from the LVI module.

Bits 0,1, 3, 6 and 7 remain unchanged. However, the HC08AZ32 device errata for bit-6 are no longer applicable and the ROMSEC bit **will** enable ROM security on the HC08AZ32A.

2.2 Mask Option Register B

This register is now located at address $FE09 (previously $003F) and has 3 new bits activated.

Bit-3, AZ32A – device indicator bit (silicon hard set bit), which identifies the new A-suffix silicon. If this bit is a ‘1’ then it is HC08AZ32A silicon.

Bit-4, EEMONSEC – EEPROM Read Protection in Monitor Mode bit. When selected as a ‘1’ the entire EEPROM array cannot be accessed in monitor mode unless a valid security code is entered.

Bit-7, EEDIVCLK – EEPROM Timebase Divider Clock Select bit which selects the reference clock source for the EEPROM timebase divider. Selected as a ‘1’ means that the CPU bus clock (possibly the PLL) drives the EEPROM time base divider. A ‘0’ selects CGMXCLK instead.

2.3 EEPROM

The HC08AZ32A EEPROM is made from a new NVM technology; however, the basic programming and erase operations remain unchanged. Bit polarity is the same; the programmed state is a logic 0 and the erased state a logic 1.

The new HC08AZ32A EEPROM requires a constant timebase source for program and erase operations. The clock source that is required to drive the EEDIV clock divider input must first be selected using the new bit-7 introduced onto the MORB register at address $FE09. Secondly, the divide ratio from this source has to be set up by programming an 11-bit time base pre-scalar into two new registers, EEDIVH and EEDIVL. These registers must be programmed with a proper value before starting any EEPROM erase or programming steps. The function of the divider is to provide a constant clock source with a period of 35µs (within ± 2µs) to the internal timer and related EEPROM circuits for proper program or erase operations. The recommended frequency range of the reference clock is 250KHz to 16MHz.

The EEDIV value is calculated by the following formula:

$$EEDIV = INT[Reference\ Frequency(Hz) \times 35 \times 10^{-6} + 0.5]$$

The result is rounded down to the nearest integer value.

For example, if the Reference Frequency is 4.9152MHz, the EEDIV value in the above formula will be 172. To examine the time base output of the divider, the Reference Frequency is divided by the calculated EEDIV value (172), which equals to 28.577KHz in frequency or 34.99µs in period.

The user must exercise caution when setting up the divide ratio – EEDIVH and EEDIVL are volatile registers. They have duplicate non-volatile registers, EEDIVHNVR and EEDIVLNVR whose contents are loaded into EEDIVH and EEDIVL upon reset. However, the user should remember to correctly set up the EEDIVH and EEDIVL registers **before** attempting to program the EEDIVHNVR and EEDIVLNVR non-volatile registers. The user has 2 options listed below.

Option 1:

1. Write the required divider value into EEDIVH and EEDIVL.
2. Call the EEPROM programming routine and program EEDIVHNVR and EEDIVLNVR with the divider value that the user would like downloaded into EEDIVH and EEDIVL every time the device is reset.

Option 2:

1. In the user's initialisation routine that is called every time the device is reset and before any EEPROM program or erase operations are attempted, write the required divider value into EEDIVH and EEDIVL
2. Ignore the non-volatile EEDIVHNVR and EEDIVLNVR registers. After a reset, the initialisation routine will be executed and the required divider value will be written into EEDIVH and EEDIVL. This will overwrite the default value of $FF that was downloaded upon reset from EEDIVHNVR and EEDIVLNVR.

The EEDIVH and EEDIVL registers are shown below and it should also be noted that Bit-7, EEDIVSECD, of EEDIVH (and EEDIVHNVR) controls EEPROM security. If this bit is programmed to 0 after system reset the security feature is permanently enabled and the divider value in the EEDIV registers cannot be changed.

EEDIVH	Bit-7	6	5	4	3	2	1	0
$FE1A	**EEDIVSECD**					EEDIV10	EEDIV9	EEDIV8
Reset:	EEDIVHNVR	X	X	X	X	EEDIVHNVR	EEDIVHNVR	EEDIVHNVR

EEDIVL	Bit-7	6	5	4	3	2	1	0
$FE1B	EEDIV7	EEDIV6	EEDIV5	EEDIV4	EEDIV3	EEDIV2	EEDIV1	EEDIV0
Reset:	EEDIVLNVR	EEDIVLNVR	EEDIVLNVR	EEDIVLNVR	EEDIVLNVR	EEDIVLNVR	EEDIVLNVR	EEDIVLNVR

NOTE: *As a result of the new EEDIV clock described above, bit 7 (EEBCLK) of the EEPROM control register (EECR) is no longer used.*

The other main difference in the HC08AZ32A EEPROM is the inclusion in the EEPROM control register (EECR) of an AUTO function. Setting bit-1 (previously unused) of that register enables the AUTO function. The AUTO function allows the logic of the MCU to automatically use the optimum programming or erasing time for the EEPROM. Using the AUTO function means that the user does not need to wait for the normal minimum specified programming or erasing time. After setting the EEPGM bit as normal, the user only has to poll that bit again, waiting for the MCU to clear it indicating that programming or erasing is complete.

Finally, the HC08AZ32A has a special feature that designates the 16 bytes of addresses from $08F0 to $08FF to be permanently secured. This security option is enabled by programming the EEPRTCT bit in the EEPROM Non-Volatile Register (EENVR, address $FE1C)) to a logic 0. Once the EEPRTCT bit is programmed to 0 for the first time programming and erasing of secured locations $08F0 to $08FF is permanently disabled. Secured locations $08F0 to $08FF can, however, be read as normal. Programming and erasing of EENVR is permanently disabled and bulk and block erase operations are disabled for the unprotected locations ($0800–$08EF and $0900–$09FF). Single byte program and erase operations are still available for locations $0800–$08EF and $0900–$09FF for all bytes that are not protected by the EEPROM Block Protect, EEPBx, bits in EENVR.

NOTE: *Once armed, the protect option is permanently enabled. Consequently, all functions in the EENVR will remain in the state they were in immediately before the security was enabled.*

2.4 Analogue to Digital Converter

The user is no longer required to select 15-channels versus 8-channels at ROM submission. Every HC08AZ32A will be configured with a 15-channel analog to digital converter.

CAUTION: *The pins used for ADC channels 12 and 14 share their functions with timer clock inputs as well as general purpose I/O. Therefore, do not use channels 14 or 12 if using TACLK or TBCLK pins as the clock inputs for the 16-bit timers.*

2.5 Timer Interface Module A

TIMA is now a 6 channel timer (previously 4 channel). Additional vectors are located at $FFCC – $FFCF which used to be ROM area.

2.6 ROM

As a result of the extra 2 timer channels described above the total amount of ROM bytes available is now 32,256 bytes (previously 32,272 bytes). The ROM on the HC08AZ32A is located at addresses $8000 – $BFFF and $C000 – $FDFF.

2.7 Monitor ROM

Increased in size to 320 bytes but functionality remains unchanged.

HC08AZ32 – 224 bytes – located at $FE20 – $FEFF
HC08AZ32A – 320 bytes – located at $FE20 – $FF5F

3 Step by step Guide

The flowchart shown in figure 1 is a step by step checklist for the user to go through and should ensure all differences between the HC08AZ32 and the HC08AZ32A have been considered before submitting a new ROM code.

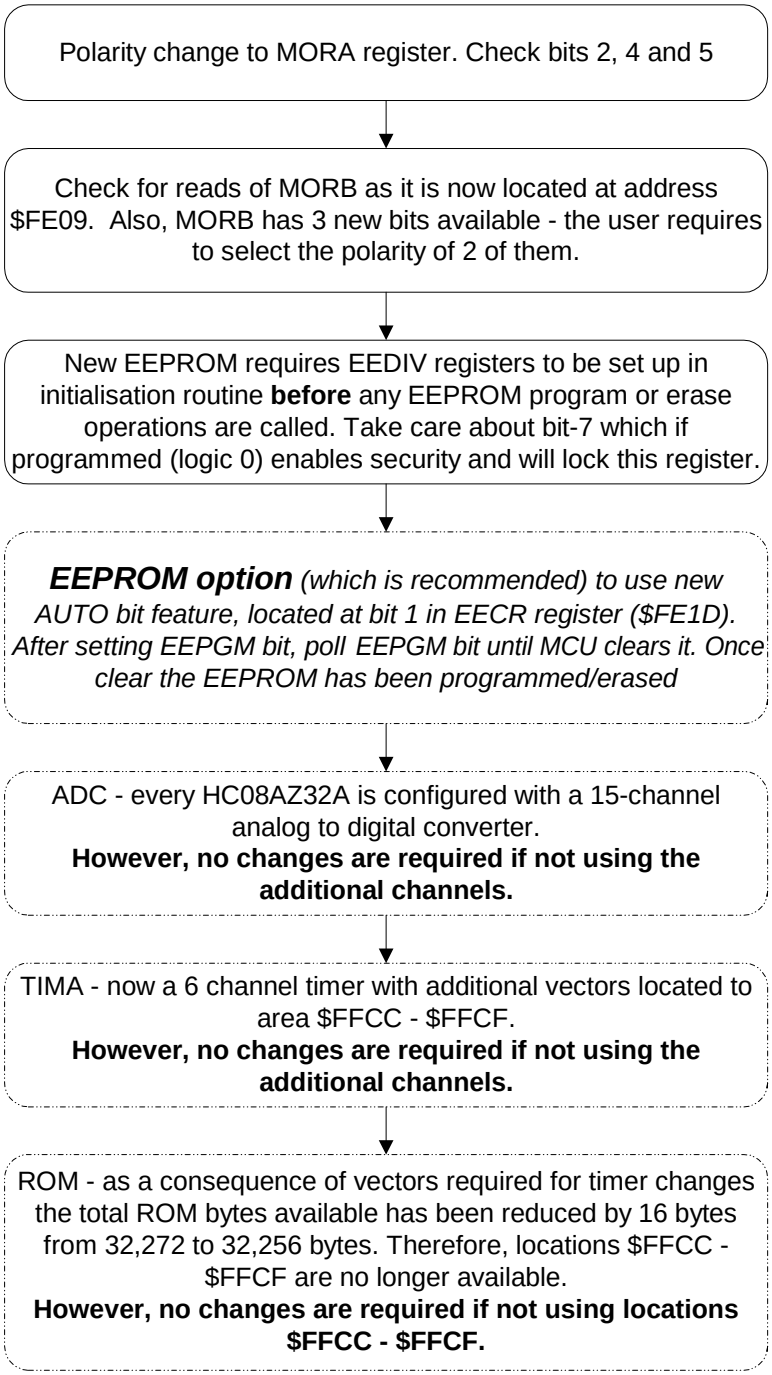

Figure 1. Step by Step Conversion Checklist

4 Conclusion

All of the differences discussed above should be checked to see if they impact the conversion of HC08AZ32 code, in particular, the Mask Option Register changes and the EEPROM changes.

Care should be taken when setting up the EEDIVH, EEDIVL, EEDIVHNVR and EEDIVLNVR registers to ensure that setting bit-7, EEDIVSECD, does not permanently enable the EEDIV security feature. This is essential if the EEDIV register values require to be changed.

It is recommended that the required divider value is written into the EEDIV registers by writing to EEDIVL first, then EEDIVH, taking care over the value written to bit-7, EEDIVSECD of EEDIVH.

Also, it is important to note that the EEDIVH and EEDIVL registers **must** be written with the required divider value before attempting to program or erase the EEPROM (including the no-volatile registers) to prevent the EEPROM from being severely damaged.

Finally, the user is advised to read the relevant chapters in the HC08AZ32A specification to ensure all differences have been fully captured. It is also advisable to gain familiarity with the HC908AZ60A emulator device, which uses the same EEPROM module.

Appendix A – EEPROM Program and Erase Algorithms

Programming

The EEPROM can be programmed such that one or multiple bits are programmed (written to a logic 0) at a time. However**, the user must not program one bit more than once before erasing the entire byte.** That is, the user is not allowed to program a logic 0 to a bit that is already programmed (bit state is already logic 0).

The unprogrammed state is a logic 1. Programming changes the state to a logic 0. Only valid EEPROM bytes in the non-protected blocks and EENVR can be programmed.

Follow the procedure below to program a byte of EEPROM after first ensuring the block protect feature is not set on the address block of the byte to be programmed:

1. Clear EERAS1 and EERAS0 and set EELAT in the EECR. (See note A)

NOTE: *If using the AUTO mode also set the AUTO bit during Step 1.*

2. Write the desired data to any user EEPROM address. (See note B)
3. Set the EEPGM bit. (See note C.) Go to step 7 if AUTO is set.
4. Wait for time, t_{EEPGM}, to program the byte.
5. Clear EEPGM bit.
6. Wait for time, t_{EEFPV}, for the programming voltage to fall. Go to step 8.
7. Poll the EEPGM bit until it is cleared by the internal timer. (See note D.)
8. Clear EELAT bits. (See note E.)

NOTE:

a. EERAS1 and EERAS0 must be cleared for programming. Otherwise, the part will be in erase mode. Setting the EELAT bit configures the address and data buses to latch data for programming the array. Only data with valid EEPROM address will be latched. If EELAT is set, other writes to the EECR will be allowed after a valid EEPROM write.

b. If more than one valid EEPROM write occurs, the last address and data will be latched overriding the previous address and data. Once data is written to the desired address, do not read EEPROM locations other than the written location. (Reading an EEPROM location returns the latched data and causes the read address to be latched).

c. The EEPGM bit cannot be set if the EELAT bit is cleared or a non-valid EEPROM address is latched. This is to ensure proper programming sequence. Once EEPROM is set, do not read any EEPROM locations; otherwise, the current program cycle will be unsuccessful. When EEPGM is set, the on-board programming sequence will be activated.

d. The delay time for the EEPGM bit to be cleared in AUTO mode is less than t_{EEPGM}. However, on other MCUs, this delay time may be different. For forward compatibility, software should not make any dependency on this delay time.

e. Any attempt to clear both EEPGM and EELAT bits with a single instruction will only clear EEPGM. This is to allow time for removal of high voltage from the EEPROM array.

Erasing

The unprogrammed state is a logic 1. Erasing changes the state of a programmed bit (logic 0) to a logic 1. Only EEPROM bytes in the non-protected blocks and EENVR can be erased. Use the following procedure to erase a byte, block or the entire EEPROM array:

1. Configure EERAS1 and EERAS0 to select byte, block or bulk erase; set EELAT in EECR. (See note A.)

NOTE: *If using the AUTO mode, also set the AUTO bit during Step 1.*

2. Byte erase: write any data to the desired address. Block erase: write any data to an address within the desired block. Bulk erase: write any data to an address within the desired array. (See note B)
3. Set the EEPGM bit. (See note C) Go to step 7 if AUTO is set.
4. Wait for a time: t_{EEBYTE} for byte erase; $t_{EEBLOCK}$ for bulk erase; t_{EEBULK} for bulk erase.
5. Clear EEPGM bit.
6. Wait for a time, t_{EEFPV}, for the erasing voltage to fall. Go to step 8.
7. Poll the EEPGM bit until it is cleared by the internal timer. (See note D.)
8. Clear EELAT bits. (See note E.)

NOTE:

a. Setting EELAT bit configures the address and data buses to latch data for erasing the array. Only valid EEPROM addresses will be latched. If EELAT is set, other writes to the EECR will only be allowed after a valid EEPROM write.

b. If more than one valid EEPROM write occurs, the last address and data will be latched overriding the previous address and data. Once data is written to the desired address, do not read EEPROM locations other than the written location. (Reading an EEPROM location returns the latched data and causes the read address to be latched).

c. The EEPGM bit cannot be set if the EELAT bit is cleared or a non-valid EEPROM address is latched. This is to ensure proper erasing sequence. Once EEPGM is set, do not read any EEPROM locations; otherwise, the current erase cycle will be unsuccessful.

d. The delay time for the EEPGM bit to be cleared in AUTO mode is less than t_{EEBYTE}/$t_{EEBLOCK}$ /t_{EEBULK}. However, on other MCUs, this delay time may be different. For forward compatibility, software should not make any dependency on this delay time.

e. Any attempt to clear both EEPGM and EELAT bits with a single instruction will only clear EEPGM. This is to allow time for removal of high voltage from the EEPROM array.

Memory Characteristics

Characteristic	Symbol	Min	Max	Unit
EEPROM Programming Time per Byte	t EEPGM	10	—	ms
EEPROM Erasing Time per Byte	t EEBYTE	10	—	ms
EEPROM Erasing Time per Block	t EEBLOCK	10	—	ms
EEPROM Erasing Time per Bulk	t EEBULK	10	—	ms
EEPROM Programming Voltage Discharge Period	t EEFPV	100	—	µs
Number of Programming Operations to the same EEPROM Byte Before Erase (1)	—	—	8	—
EEPROM Write/Erase Cycles @ 10 ms Write Time +125 °C	—	10,000	—	Cycles
EEPROM Data Retention After 10,000 Write/Erase Cycles	—	10	—	Years
EEPROM Programming Maximum Time to ‘AUTO’ Bit Set	—	—	500	µs
EEPROM Erasing Maximum Time to ‘AUTO’ Bit Set	—	—	8	ms

1. Programming a byte more times than the specified maximum may affect the data integrity of that byte. The byte must be erased before it can be programmed again.

How to Reach Us:

Home Page:
www.freescale.com

E-mail:
support@freescale.com

USA/Europe or Locations Not Listed:
Freescale Semiconductor
Technical Information Center, CH370
1300 N. Alma School Road
Chandler, Arizona 85224
+1-800-521-6274 or +1-480-768-2130
support@freescale.com

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
support@freescale.com

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064
Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
1-800-441-2447 or 303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com