# AN-HK-31

# Using the MC68HC908GP32 in place of MC68HC908GP20

By Lewis Lai
Microcontroller Division
Hong Kong



This application note describes the differences between the MC68HC908GP32 (GP32) and the MC68HC908GP20 (GP20) microcontroller units. Furthermore, it highlights the areas where the user must consider if the GP32 is to be used in existing applications using the GP20.

The text is divided into three parts. Part 1 describes the differences between the two devices. Part 2 discusses the change from GP20 to GP32. Finally, part 3 provides information on additional documentation and development tools for both devices.

The timings and component values in this application note are extracted from the revision 2 datasheets. For detailed specifications on both devices, the latest datasheet should be consulted. Use order numbers MC68HC908GP20/D and MC68HC908GP32/H for the GP20 and the GP32 datasheets respectively.

## Introduction

The GP32 MCU is an extended version of the GP20. It is pin-to-pin compatible with the GP20; therefore, it can be used as a replacement in existing GP20 applications.

Although the GP32 can be used in an GP20 application, there are some distinct differences between the two devices. They are:

- GP32 has 32k-bytes of FLASH memory; GP20 has 20k-bytes.
- Program and erase algorithms for the FLASH memory are different for each device.

## PART 1 — GP32 and GP20: A Comparison

### Memory map and register locations

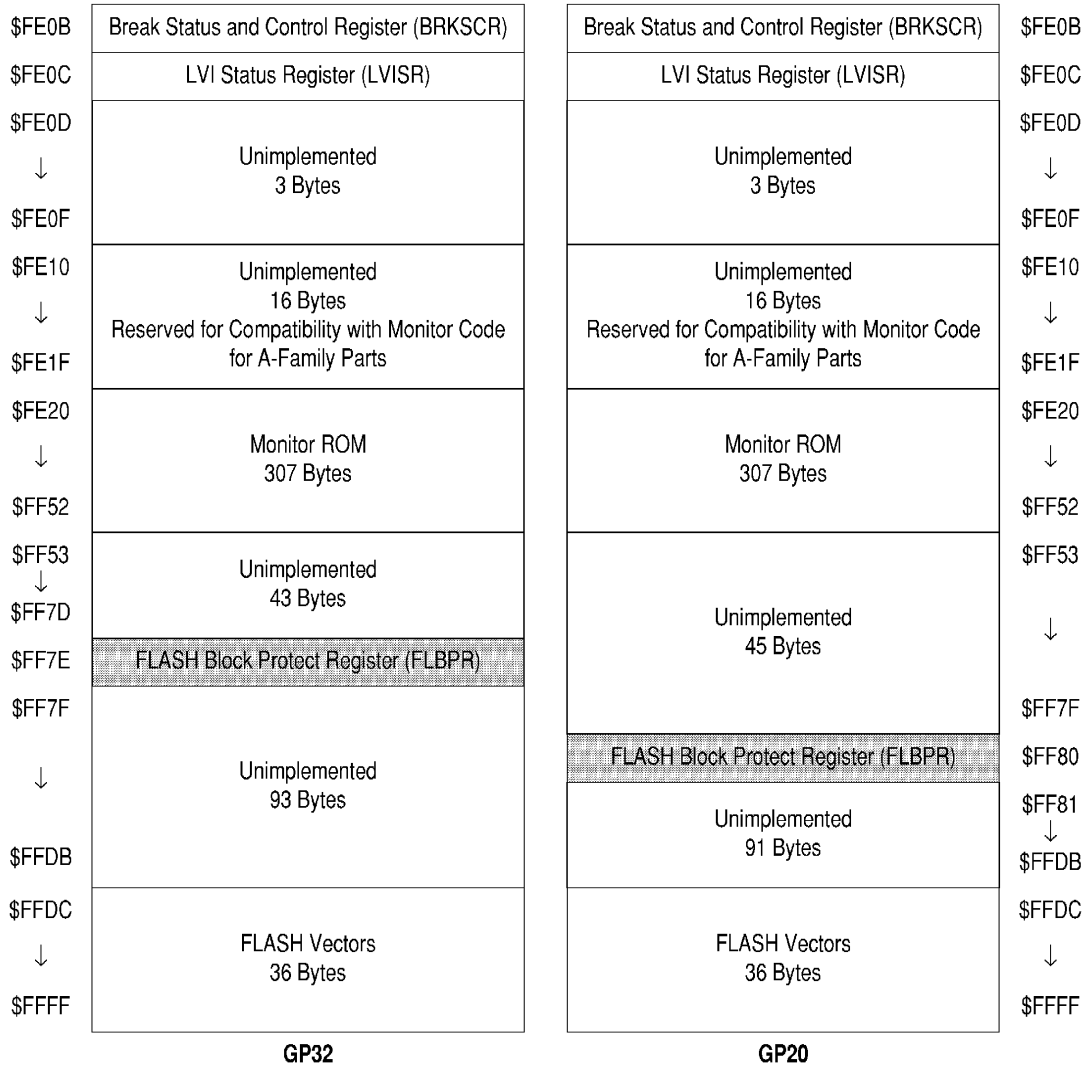
Table 1 shows the memory maps for the GP32 and GP20. The shaded areas indicate the differences between the two devices. They are:

- Start of FLASH memory area.
- Location of FLASH block protect register.

**Table 1. GP32 vs. GP20 Memory Map**

| | GP32 | GP20 | |
|---|---|---|---|
| $0000 ↓ $003F | I/O Registers 64 Bytes | I/O Registers 64 Bytes | $0000 ↓ $003F |
| $0040 ↓ $023F | RAM 512 Bytes | RAM 512 Bytes | $0040 ↓ $023F |
| $0240 ↓ $7FFF | Unimplemented 32,192 Bytes | Unimplemented 44,480 Bytes | $0240 ↓ $AFFF |
| $8000 ↓ $FDFF | FLASH Memory 32,256 Bytes | FLASH Memory 19,968 Bytes | $B000 ↓ $FDFF |
| $FE00 | SIM Break Status Register (SBSR) | SIM Break Status Register (SBSR) | $FE00 |
| $FE01 | SIM Reset Status Register (SRSR) | SIM Reset Status Register (SRSR) | $FE01 |
| $FE02 | Reserved (SUBAR) | Reserved (SUBAR) | $FE02 |
| $FE03 | SIM Break Flag Control Register (SBFCR) | SIM Break Flag Control Register (SBFCR) | $FE03 |
| $FE04 | Interrupt Status Register 1 (INT1) | Interrupt Status Register 1 (INT1) | $FE04 |
| $FE05 | Interrupt Status Register 2 (INT2) | Interrupt Status Register 2 (INT2) | $FE05 |
| $FE06 | Interrupt Status Register 3 (INT3) | Interrupt Status Register 3 (INT3) | $FE06 |
| $FE07 | Reserved | Reserved | $FE07 |
| $FE08 | FLASH Control Register (FLCR) | FLASH Control Register (FLCR) | $FE08 |
| $FE09 | Break Address Register High (BRKH) | Break Address Register High (BRKH) | $FE09 |
| $FE0A | Break Address Register Low (BRKL) | Break Address Register Low (BRKL) | $FE0A |

**Table 1. GP32 vs. GP20 Memory Map**




**Size of FLASH memory**

GP32 has a larger FLASH memory than the GP20.

- GP32 has 32,256 bytes of FLASH.
- GP20 has 19,968 bytes of FLASH.

**Erased state of the FLASH memory**

The erased state of the FLASH memory is different between the two devices.

- GP32: an erased byte reads as $FF.
- GP20: an erased byte reads as $00.

## FLASH memory block protection

Both the GP20 and GP32 has the ability to protect its FLASH memory from unintentional erase or program operations due to system malfunction. The protection mechanism on both devices is controlled by a FLASH block protect register.

This FLASH block protect register is located at $FF80 for GP20, and $FF7E for GP32; the control methods are not the same. These two registers are described in the following paragraphs.

## GP20 FLASH block protection

The GP20 block protection uses four bits in the block protect register.

| | | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|
| Address: | Read: | Not used | | | | BPR3 | BPR2 | BPR1 | BPR0 |
| $FF80 | Write: | | | | | | | | |
| | Reset: | U | U | U | U | U | U | U | U |

U = Unaffected by reset. Initial value from factory is 0.
Write to this register is by a programming sequence to the FLASH memory.

**Figure 1. GP20 FLASH Block Protect Register**

There are two protection options for the GP20 FLASH memory.

BPR3 — Setting this bit protects the FLASH memory range from $C000 to $FFFF.

BPR2, BPR1, and BPR0 — Setting any of these bits will protect the entire 20k-byte FLASH memory, from $B000 to $FFFF.

## GP32 FLASH block protection

Compared with the GP20, the GP32 has a more flexible FLASH protection mechanism.

| | | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|
| Address: | Read: | BPR7 | BPR6 | BPR5 | BPR4 | BPR3 | BPR2 | BPR1 | BPR0 |
| $FF7E | Write: | | | | | | | | |
| | Reset: | U | U | U | U | U | U | U | U |

U = Unaffected by reset. Initial value from factory is 1.
Write to this register is by a programming sequence to the FLASH memory.

**Figure 2. GP32 FLASH Block Protect Register**

The eight bits in this register represent bits [14:7] of a 16-bit memory address. Bit 15 is always 1 (FLASH starts at $8000), and bits [6:0] are 0's. This is illustrated in figure 3 below.

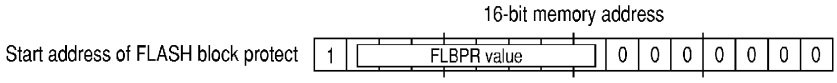



**Figure 3. FLASH Block Protect Start Address**

The resultant 16-bit address is used for specifying the start address of the FLASH memory for block protection. The FLASH is protected from this start address to the end of FLASH memory, at $FFFF. With this mechanism, the protect start address can be XX00 and XX80 (128-byte boundaries) within the FLASH memory.

Examples of protect start address:

| BPR[7:0] | Start of Address of Protect Range |
|---|---|
| $00 | The entire FLASH memory is protected. |
| $01 (**0000 0001**) | $8080 (1**000 0000 1**000 0000) |
| $02 (**0000 0010**) | $8100 (1**000 0001 0**000 0000) |
| and so on... | |
| $FE (**1111 1110**) | $FF00 (1**111 1111 0**000 0000) |
| $FF | The entire FLASH memory is not protected. |

Note:
The end address of the protected range is always $FFFF.

## Optimum FLASH programming bus frequency

Both the GP20 and GP32 uses an internal charge pump to generate the high voltage for programming the FLASH memory. The charge pump clock is derived from the bus clock in the MCU.

On the GP20, the minimum bus clock frequency that is required for successful FLASH program or erase is 2MHz. Higher bus frequencies require configuring the FDIV1 and FDIV0 bits in the FLASH control register for FLASH program or erase.

On the GP32, the minimum bus clock frequency that is required for successful FLASH program or erase is 1MHz. Higher bus frequencies does not require any register configuration. This is managed automatically inside the MCU.

## Charge-pump ON/OFF

In the CONFIG2 register (address $001E) on GP20, the PMPSGLVEN bit (bit 2) controls the on/off state of the internal charge-pump. The default setting is 0 (on), for a 5V operating voltage. This bit needs to be set to 1 when the operating voltage is 3V.

On the GP32, there is no PMPSGLVEN bit. The control voltage to the charge pump is regulated automatically. Bit 2 in CONFIG2 register is not implemented, and will always read as 0.

## FLASH memory erasing and programming

Programming tools are available from Motorola for programming and erasing the FLASH memory. These are always recommended; as these tools are fully proven and tested for optimum FLASH programming and erasing. Part numbers for these tools are listed in part 3 of this application note.

For users wishing to write their own routines for program or erase, the FLASH control register at address $FE08 is used for this purpose. Although the FLASH control register is at the same address on both devices, the program/erase algorithms are different (see figures 6 and 7). Also, despite some register bits have the same name, their uses are different (see figure 4 and 5). For full details on program and erase operations, the datasheets should be consulted. The following paragraphs briefly describe the FLASH program and erase capabilities.

| | | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|
| Address: $FE08 | Read: / Write: | FDIV1 | FDIV0 | BLK1 | BLK0 | HVEN | MARGIN | ERASE | PGM |
| | Reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 4. GP20 FLASH Control Register**

| | | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|
| Address: $FE08 | Read: | 0 | 0 | 0 | 0 | HVEN | MASS | ERASE | PGM |
| | Write: | | | | | | | | |
| | Reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented

**Figure 5. GP32 FLASH Control Register**

## GP20 FLASH erasing and programming

First, lets clarify some terms on the GP20 FLASH.

- A page consists of 8 consecutive bytes. This is the maximum size of FLASH that can be programmed in a program sequence.
- A row consists of 8 pages.
- A block is the size of FLASH to be erased or it can refer to a range of protected FLASH memory. The minimum size of FLASH that can be erased in an erase sequence is a single row, i.e. 64 bytes.

The erase block size of the GP20 can be 64 bytes, 512 bytes, 4k-bytes, 16k-bytes, or 20k-bytes. This is selected by the BLK1 and BLK0 bits.

The GP20 FLASH is programmed in 8-byte (page) bursts; with the sequence repeated until all required locations are programmed. The margin read bit, MARGIN, is used to ensure programmed bits are programmed to sufficient margin for data retention over the device lifetime. Therefore, a page is programmed repeatedly until margin read data equals write data. This MARGIN bit is not available in the GP32 FLASH control register — it is not required. Figure 6 shows the GP20 programming algorithm, extracted from the datasheet.

**GP32 FLASH erasing and programming**

As again, lets clarify two terms on the GP32 FLASH.

- A row consists of 64 consecutive bytes. This is the maximum size of FLASH that can be programmed in a program sequence.
- A page consists of 128 consecutive bytes. This is the minimum size of FLASH that can be erased in an erase sequence.

The GP32 FLASH supports two types of erase: page-erase and mass- or bulk-erase. A page-erase erases 128 bytes of FLASH, whereas a mass-erase erases the entire 32k-byte FLASH memory. The MASS bit is set for mass-erase operations.

The GP32 FLASH is programmed in 64-byte (row) bursts; with the sequence repeated until all required locations are programmed. Unlike the GP20, margin read is not required for the GP32 FLASH to ensure sufficient programming of the cells. Figure 7 shows the GP32 programming algorithm, extracted from the datasheet.

**FLASH programming times**

From figures 6 and 7, the programming times can be calculated for the two devices.

- 62.4 seconds are required to program the 20k-bytes on GP20.
- 0.978 seconds are required to program the 32k-bytes on GP32.

These typical timings are for comparison only. The actual programming time would need to include instruction cycle times, communication times for data download from host to MCU, and other related operations required for setting up the programming sequences.

**Program/erase endurance**

The number of program or erase cycles for the FLASH memory are:

- 100 program/erase cycles for GP20.
- 10000 program/erase cycles for GP32.

Programming time for the 20k FLASH array on GP20

$$= (t_{PROG} + t_{HVTV} + t_{VTP} + t_{HVD}) \times N_{ATTEMPT} \times \frac{19{,}968}{8}$$

$$= (1000 + 50 + 150 + 50)\ \mu s \times 20 \times 2496$$

$$= 62.4\ s$$

Programming time for the 36 bytes user vectors on GP20

$$= (t_{PROG} + t_{HVTV} + t_{VTP} + t_{HVD}) \times N_{ATTEMPT} \times \frac{36}{8}$$

$$= (1000 + 50 + 150 + 50)\ \mu s \times 20 \times 5$$

$$= 125\ ms$$

Where $N_{ATTEMPT}$ is the typical number of attempts required to program a page.

**Algorithm for programming a page (8 bytes) of FLASH memory in GP20**

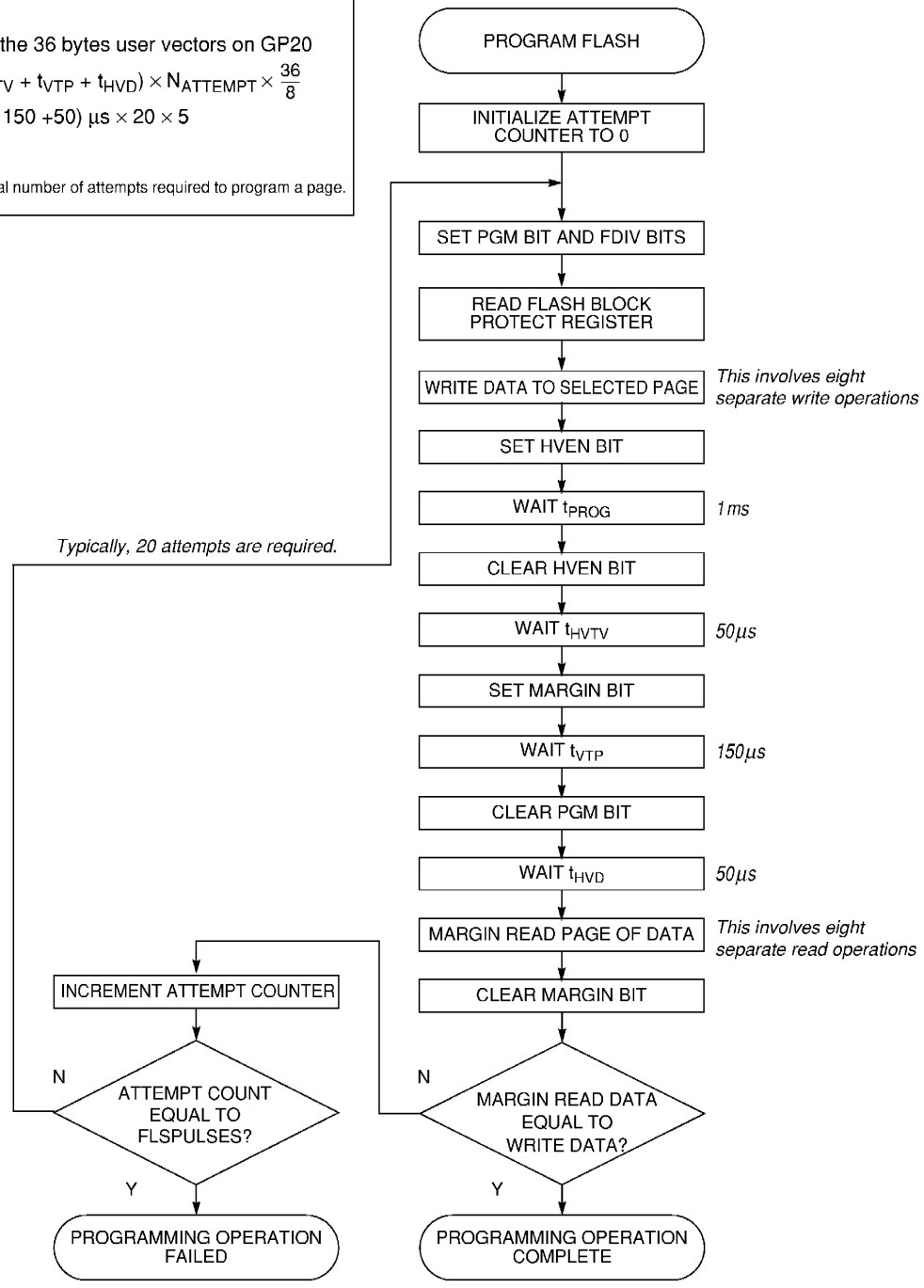



**Figure 6. GP20 Programming Algorithm**

**Algorithm for programming a row (64 bytes) of FLASH memory in GP32**

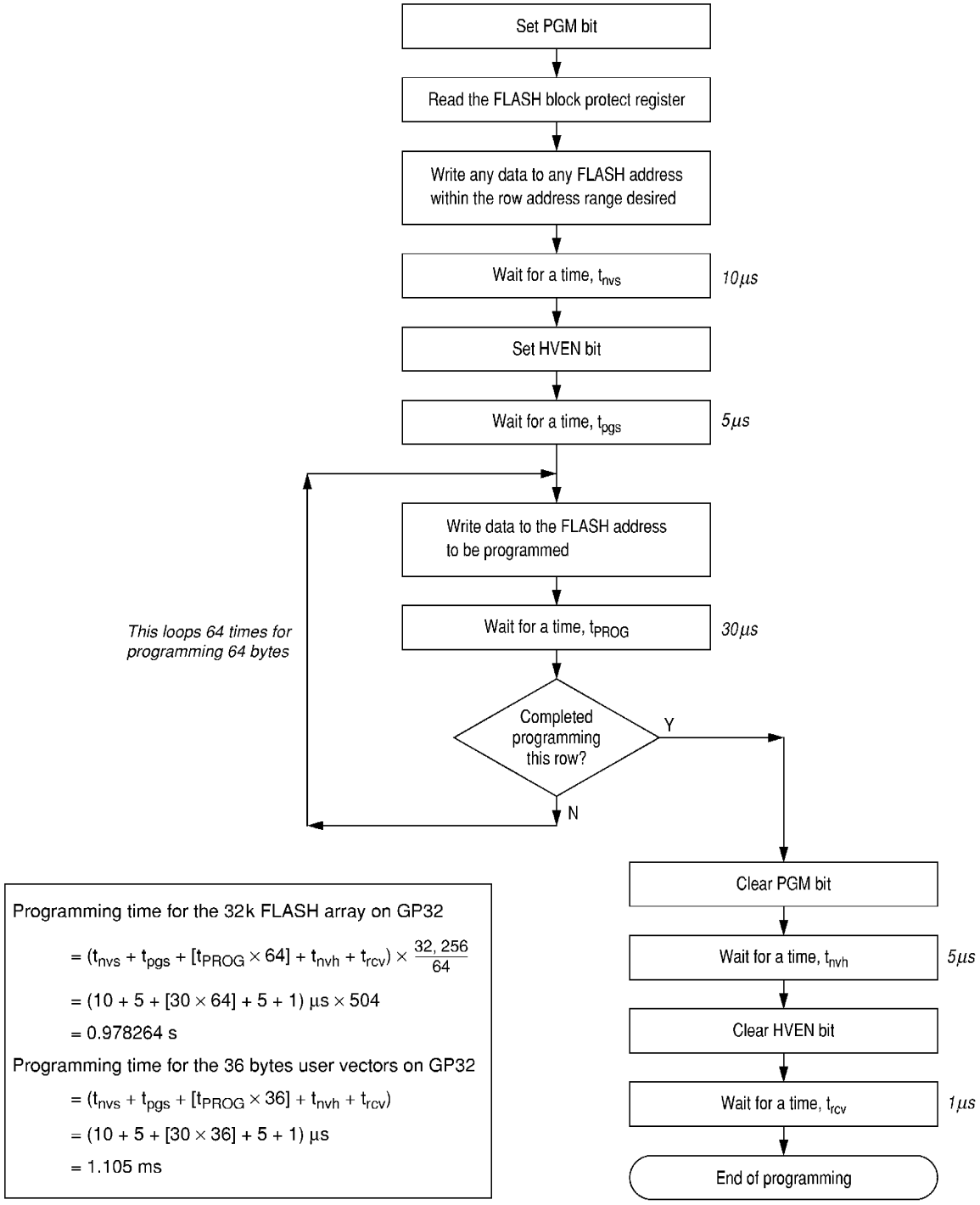



**Figure 7. GP32 Programming Algorithm**

## PART 2 — From GP20 to GP32: Some Considerations

Part 1 described the differences between the two devices. This part of the application note will highlight some points for users wishing to use the GP32 to replace the GP20 in their applications.

### User Code

The user code on the GP20 can be used, without modification, on the GP32 if the following conditions are satisfied:

- The user code is programmed to the same area of FLASH memory in the GP32 as in the GP20, leaving $8000 to $AFFF unused.
- The user code does not access the FLASH control register at $FE08.
- The user code does not access the FLASH block protection register at either $FF7E or $FF80.
- The user code does not use data from CONFIG2 register bit-2 for any control purposes.

### In-circuit programming the FLASH

In-circuit programming (ICP) refers to the programming of FLASH memory when the MCU is on the final circuit board.

User code modification is not required if the FLASH registers are not accessed during run-time. In this case, ICP is carried out using a serial link between a host system and the final circuit board, with the MCU running in monitor mode.

If the run-time user code access the FLASH registers to re-program part, or the entire FLASH, the user code will need modification.

Further details on ICP for the two devices can be found documented in their application notes on ICP (see part 3 for order numbers).

### The PMPSGLVEN bit

Bit 2 of CONFIG2 (address $001E) is not implemented on the GP32; and it will always read as 0. If the user code sets this bit in GP20 to disable the voltage regulator to the charge-pump, this bit will always read as 1 during run-time. The user must be aware if this bit is read during run-time for other control purposes.

### CGM external components

Please refer to the respective datasheets for the detailed discussion on the Clock Generation Module.

## Port A, C, and D pull-up resistors

For GP32, the pull-up resistor on each of the port A, C, and D pins (as inputs) is typically 45kΩ. For GP20, it is typically 33kΩ. Therefore, the current drawn by the pins is different between the two devices. The following table is the data extracted from the datasheets.

| | | | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|
| Pull-up resistors (as input only) Ports PTA7/$\overline{\text{KBD7}}$–PTA0/$\overline{\text{KBD0}}$, PTC6–PTC0, PTD7/T2CH1–PTD0/$\overline{\text{SS}}$ | $R_{PU}$ | GP32 | 20 | 45 | 65 | kΩ |
| | | GP20 | 20 | 33 | 50 | kΩ |

## LVI voltage trip- points

Examining the datasheets also show that there are minor variations in the LVI trip-point voltages. Users may need to take this into consideration when changing to GP32.

| | | $V_{DD}$ = 5V | | | $V_{DD}$ = 3V | | | |
|---|---|---|---|---|---|---|---|---|
| | | Min. | Typ. | Max. | Min. | Typ. | Max. | Unit |
| $V_{TRIPF}$ | GP32 | 3.90 | 4.25 | 4.50 | 2.45 | 2.60 | 2.70 | V |
| | GP20 | 4.13 | 4.30 | 4.35 | 2.50 | 2.60 | 2.63 | V |
| $V_{TRIPR}$ | GP32 | 4.20 | 4.35 | 4.60 | 2.55 | 2.66 | 2.80 | V |
| | GP20 | 4.23 | 4.40 | 4.45 | 2.60 | 2.66 | 2.73 | V |
| $V_{HYS}$ | GP32 | — | 100 | — | — | 60 | — | mV |
| | GP20 | — | 100 | — | — | 60 | — | mV |

## PART 3 — Additional Information

The following parts can be ordered from Motorola or our representatives.

### Technical Documents

The following documents can be ordered from Motorola, or downloaded directly from the World-Wide-Web site at http://www.mcu.motsps.com.

***MC68HC908GP20/D***
MC68HC908GP20 Technical Databook.

***MC68HC908GP32/H***
MC68HC908GP32 Technical Databook.

***AN1770/D***
Application Note: In-Circuit Programming of FLASH Memory in the MC68HC908GP20.

***AN-HK-32/H***
Application Note: In-Circuit Programming of FLASH Memory in the MC68HC908GP32.

## Development tools

The following emulation and programming tools are available from Motorola for the GP20 and GP32.

**EMULATOR PART NUMBERS: choose 1 + 2 + 3 + 4**

| GP20 | GP32 | Description | |
|---|---|---|---|
| Choose one of two emulation platform systems: | | | 1 |
| M68MMDS08 | | High performance MMDS emulator station | |
| M68MMPFB0508 | | MMEVS platform board | |
| Choose the emulator module: | | | 2 |
| M68EML08GP20 | X68EML08GP32 | Emulation daughter board | |
| Choose the target head: | | | 3 |
| M68TC08GP20P40 | M68TB08GP32P40 | 40-pin DIP target head adapter | |
| M68TC08GP20FB44 | — | 44-pin QFP target head adapter with TQsocket and TQpack | |
| M68TQS044SAG1 | — | 44-pin QFP TQsocket surface mount adapter with guides | |
| M68TQP044SAM01 | — | 44-pin QFP TQpack surface mount adapter | |
| — | M68TB08GP32B42 | 42-pin SDIP target head adapter | |
| — | M68TC08GP32FB44 | 44-pin QFP target head adapter | |
| Choose the flex-cable: | | | 4 |
| M68CBL05B (used with M68**TC**08... target adapters) | | Low noise flex-cable | |
| M68CBL05C (used with M68**TB**08... target adapters) | | Low noise flex-cable | |

**PROGRAMMER PART NUMBERS: choose A or B or C**

| | | | |
|---|---|---|---|
| Use this kit: | | | A |
| M68ICS08GP20 | M68ICS08GP32 | Programmer / in-circuit debug kit | |
| Choose the programmer platform: | | | B |
| — | M68HC705UPGMR | Universal programmer for a single device | |
| — | M68HC705UGANG | Universal gang programmer for 10 devices | |
| Choose the adapter: | | | |
| — | X68UPA08GP32P40 | 40-pin DIP programmer adapter for UPGMR | |
| — | X68UPA08GP32B42 | 42-pin SDIP programmer adapter | |
| — | X68UPA08GP32FB44 | 44-pin QFP programmer adapter | |
| Use this programmer: | | | C |
| — | M68SPGM08 | Serial programmer | |
| Choose the adapter: | | | |
| — | M68PA08GP32P40 | 40-pin DIP programmer adapter | |
| — | M68PA08GP32B42 | 42-pin SDIP programmer adapter | |
| — | M68PA08GP32FB44 | 44-pin QFP programmer adapter | |

## Notes

## Notes

### *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



Freescale Semiconductor, Inc.