GMS77C1000/1001

EPROM Programming/Verify Specification

This document includes the programming specifications for the following devices:
-GMS77C1000
-GMS77C1001

INTRODUCTION

Overview

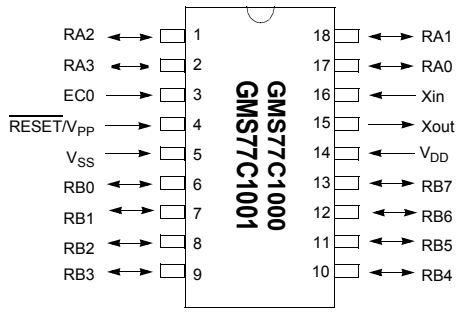
The GMS77C1000/1001 are a single-chip CMOS microcontrollers with on-chip EPROM for program storage. The programming specification also applies to ROM products for verification only.

Due to the special architecture of these microcontrollers (12-bit wide instruction word) and the low pin counts (starting at 18 pins), the EPROM programming methodology is different from that of standard (byte-wide) EPROMs (e.g., 27C256). The GMS77C1000/1001 can be programmed by applying the 12-bit wide data word to the 12 available I/O pins while the address is generated by the on-chip Program Counter. The $\overline{\text{RESET}}$/VPP pin provides the programming supply voltage (VPP). Programming/verify chip enable is controlled by the EC0 pin while the Xin pin controls the Program Counter.

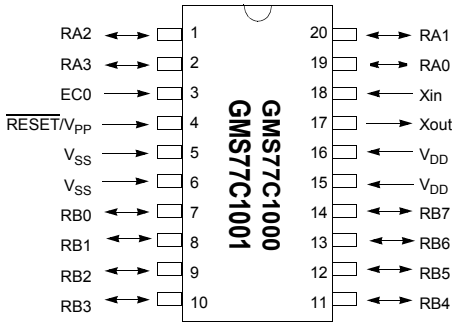
This document describes all the programming details of the GMS77C1000/1001 and the requirements for programming equipment to be used from programming prototypes in the engineering lab up to high volume programming on the factory floor.

Pin Diagrams

PDIP, SOIC



SSOP



PIN DESCRIPTIONS (DURING PROGRAMMING): GMS77C1000/1001

Pin Name	During Programming		
	Pin Name	Pin Type	Pin Description
EC0	PROG/VER	I	Program pulse input/verify pulse input
RA0 - RA3	D0 - D3	I/O	I/O Data input/output
RB0 - RB7	D4 - D11	I/O	I/O Data input/output
Xin	INCPC	I	Increment Program Counter input
$\overline{\text{RESET}}$/VPP	VPP	P	Programming Power
VDD	VDD	P	Power Supply
VSS	VSS	P	Ground

Legend: I = Input, O = Output, P = Power

1. PROGRAM/VERIFY MODES

The GMS77C1000/1001 Series uses the internal Program counter (PC) to generate the EPROM address. VPP is supplied through the $\overline{\text{RESET}}$ pin.
The EC0 pin acts as chip enable, alternating between programming and verifying.
The Xin pin is used for incrementing the PC.
Data is applied to, or can be read on PORTA andPORTB (MSB on RB7, LSB on RA0).
The programming/verify mode is entered by raising the level on the $\overline{\text{RESET}}$ pin from VIL to VHH (= VPP) while the EC0 pin is held at VIH and the Xin pin is held at VIL.
The Program Counter now has the value "0xFFF", because $\overline{\text{RESET}}$ was at VIL before. This condition selects the configuration word as the very first EPROM location to be accessed after entering the program/verify mode.
Since the $\overline{\text{RESET}}$ pin was initially at VIL, the device is in the reset state (the I/O pins are in the reset state).
Incrementing the PC once (by pulsing the Xin pin) selects location "0x000" of the user program memory.
Afterwards all other memory locations from 001h through end of memory can be addressed by incrementing the PC.
If the Program Counter has reached the last address of the user memory area (e.g. "0x1FF" for the GMS77C1000), and is incremented again, the on-chip special EPROM area will be addressed. (See Figure 1-2 to determine where the special EPROM area is located for the GMS87C1000/1001 devices).

1.1 Program/Verify Without PC Increment

After entering the program/verify mode, pulsing the EC0 pin LOW programs the data present on PORTA and PORTB into the memory location selected by the Program Counter. The duration of the EC0 LOW time determines the length of the programming pulse.
Pulsing the EC0 pin LOW again without changing the signals on $\overline{\text{RESET}}$ and Xin puts the contents of the selected memory location out on PORTA and PORTB for verification of a successful programming cycle.
This verification pulse on EC0 can be much shorter than the programming pulse. If the programming was not successful, EC0 can be pulsed LOW again to apply another programming pulse, followed again by a shorter EC0 LOW pulse for another verification cycle.
This sequence can be repeated as many times as required until the programming is successful.

1.2 Verify with PC Increment

If a verification cycle shows that programming was suc-cessful, the Program Counter can be incremented by keeping the EC0 input at a HIGH level while pulsing the Xin input HIGH. When both EC0 and Xin are HIGH, the contents of the selected memory loca-tion is put out on Ports A and B (= Verify). The falling edge of Xin will increment the Program Counter.
A fast VERIFY- ONLY with automatic increment of the PC can be performed by entering the program/verify mode as described above and then clocking the Xin input. If Xin is HIGH, the selected memory location is output on Ports A and B, while the falling edge of Xin will increment the Program Counter. Thus, the first memory location to be verified after entering the program/verify mode, is the configuration word. The next location is 000h followed by 001h and so on. The program memory location "N" can be reached by gen-erating "N + 1" falling edges on Xin. When Xin is brought HIGH again, the contents of address "N" are output on Ports A and B as long as Xin stays HIGH.

1.3 Programming/Verifying Configuration Word

The configuration word is logically mapped at program memory location "0xFFF". The PC points to the config-uration word after $\overline{\text{RESET}}$ pin goes from LOW to VHH (HIGH). The configuration word can be programmed or verified using the techniques described in Section 1.1 and Section 1.2.
If PC is incremented, the next location it will point to is "0x000" in user memory. Incrementing PC 4096 times will not allow the user to point to the configuration word.
The only way to point to it again is to reset and re-enter program mode

1.4 Programming Method

The programming technique is described in the follow-ing section. It is designed to guarantee good programming margins. It does, however, require a variable power supply for VCC.

1.4.1 PROGRAMMING METHOD DETAILS

Essentially, this technique includes the following steps:

1. Perform blank check at VDD = VDDmin.
   Report failure. The device may not be properly erased.
2. Program location with pulses and verify after each pulse at VDD = VDDP:
   where VDDP = VDD range required during programming(4.75V - 5.25V).

a) Programming condition:
   VPP = 11.5V to 12.5V
   VDD = VDDP = 4.75V - 5.25V
   VPP must be ≥ VDD + 6.25V to keep "programming mode" active.

b) Verify condition:
   VDD = VDDP
   VPP ≥ VDD + 6.5V but not to exceed 12.25V
   If location fails to program after "N" pulses, (suggested maximum program pulses of 8) then report error as a programming failure.

Note: *Device must be verified at minimum and maximum specified operating voltages as specified in the data sheet.*

3. Once location passes "Step 2", apply 11X overprogramming, i.e., apply eleven times the number of pulses that were required to program the location.
   This will guarantee a solid programming margin. The overprogramming should be made "software programmable" for easy updates.
4. Program all locations.
5. Verify all locations (using speed verify mode) at VDD = VDDmin
6. Verify all locations at VDD = VDDmax
   VDDmin is the minimum operating voltage spec. for the part. VDDmax is the maximum operating volt-age spec. for the part.

1.4.2 SYSTEM REQUIREMENTS

Clearly, to implement this technique, the most stringent requirements will be that of the power supplies:

VPP: VPP can be a fixed 11.5V to 12.5V supply. It must not exceed 13.0V to avoid damage to the pin and should be current limited to approximately 100mA.

VDD: 2.0V to 6.5V with 0.25V granularity. Since this method calls for verification at different VDD values, a programmable VDD power supply is needed.

Current Requirement: 100mA maximum

Hynix may release GMS77C1000/1001 in the future with different VDD ranges which make it necessary to have a programmable VDD.
It is important to verify an EPROM at the voltages specified in this method to remain consistent with Hynix's test screening. For example, a GMS77C1000/1001 specified for 4.75V - 5.25V should be tested for proper programming from 4.75V - 5.25V.

Note: *Any programmer not meeting the program-mable VDD requirement and the verify at VDDmax and VDDmin requirement may only be classified as "prototype" or "development" programmer but not a production programmer.*

1.4.3 SOFTWARE REQUIREMENTS

Certain parameters should be programmable (and therefore easily modified) for easy upgrade.

a) Pulse width

b) Maximum number of pulses, current limit 8.

c) Number of over-programming pulses: should be = (A • N) + B, where N = number of pulses required in regular programming. In our current algorithm A = 11, B = 0.

1.5 Programming Pulse Width

Program Memory Cells: When programming one word of EPROM, a programming pulse width (TPW) of 100 μs is recommended.
The maximum number of programming attempts should be limited to 8 per word.
After the first successful verify, the same location should be over-programmed with 11X over-programming.

Configuration Word: The configuration word for oscillator selection, WDT (watchdog timer) disable and code protection, requires a programming pulse width (TPWF) of 10 ms. A series of 100 μs pulses is preferred over a single 10 ms pulse.

FIGURE 1-1: PROGRAMMING METHOD FLOWCHART

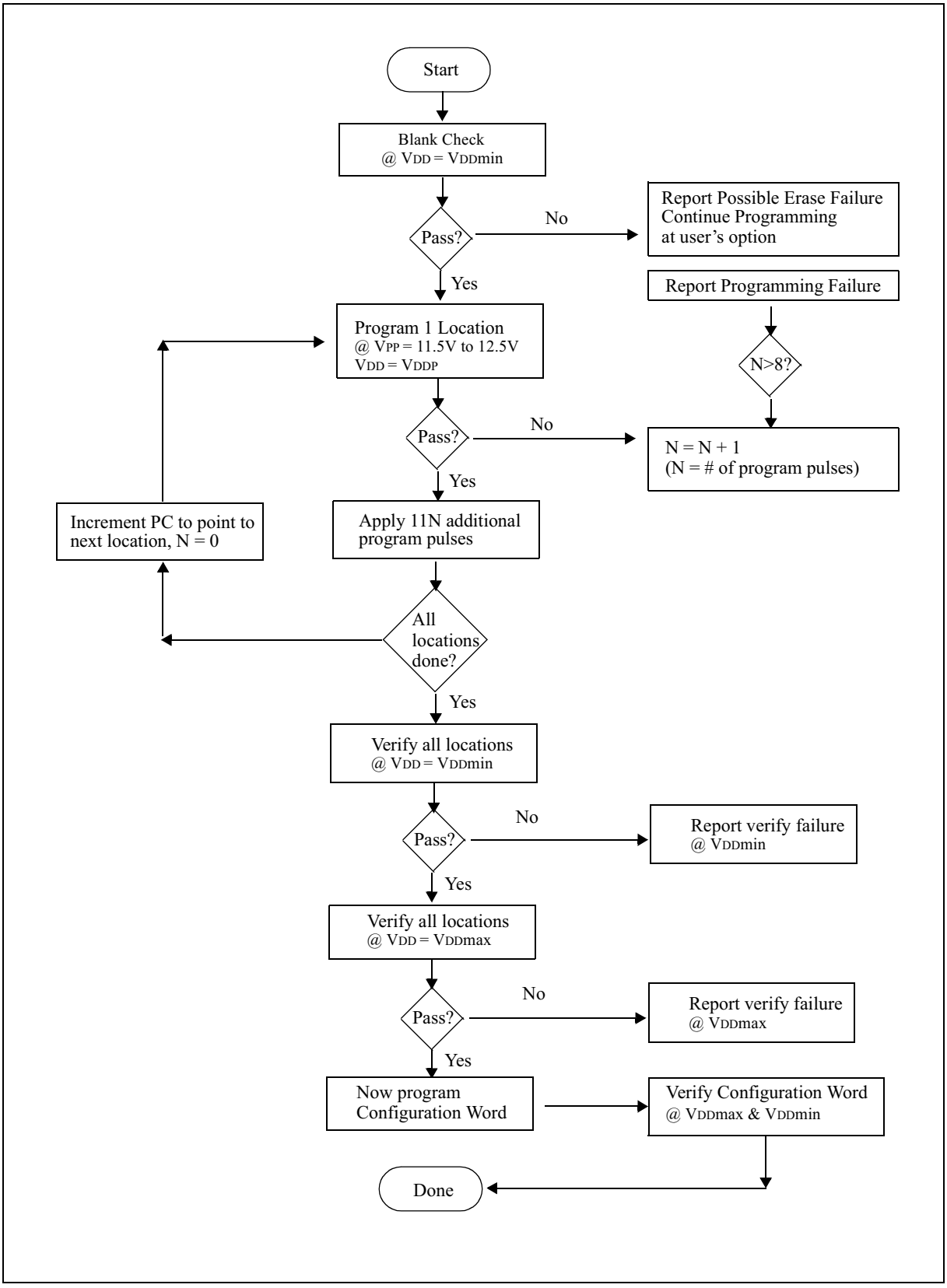
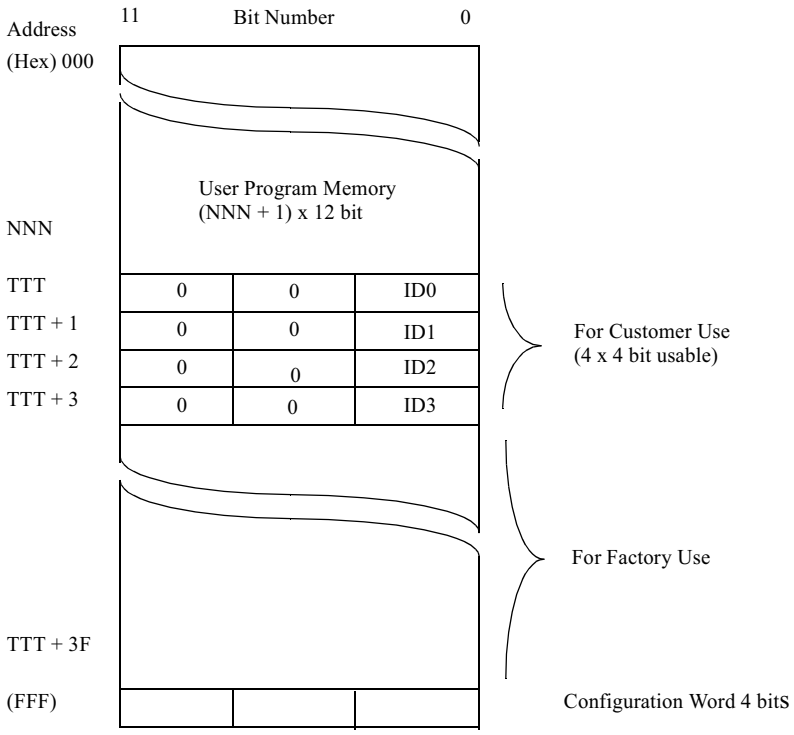


FIGURE 1-2: GMS77C1000/1001 PROGRAM MEMORY MAP IN PROGRAM/VERIFY MODE



NNN Highest normal EPROM memory address. NNN = 0x1FF for GMS77C1000.
NNN = 0x3FF for GMS77C1001.
TTT Start address of special EPROM area and ID Locations.

1.6 Special Memory Locations

The ID Locations area is only enabled if the device is in a test or programming/verify mode. Thus, in normal operation mode only the memory location 0x000 to 0xNNN will be accessed and the Program Counter will just roll over from address 0xNNN to 0x000 when incremented.

The configuration word can only be accessed immediately after $\overline{\text{RESET}}$ going from V_{IL} to V_{HH}. The Program Counter will be set to all ’1’s upon $\overline{\text{RESET}}$ =V_{IL}. Thus, it has the value “0xFFF” when accessing the configura-tion EPROM. Incrementing the Program Counter once by pulsing Xin causes the Program Counter to roll over to all '0's. Incrementing the Program Counter 4K times after reset ($\overline{\text{RESET}}$ = V_{IL}) does not allow access to the configuration EPROM.

1.6.1 CUSTOMER ID CODE LOCATIONS

Per definition, the first four words (address TTT to TTT + 3) are reserved for customer use. It is recommended that the customer use only the four lower order bits (bits 0 through 3) of each word and filling the eight higher order bits with '0's.

A user may want to store an identification code (ID) in the ID locations and still be able to read this code after the code protection bit was programmed. This is possible if the ID code is only four bits long per memory location, is located in the least significant nibble boundary of the 12-bit word, and the remaining eight bits are all '0's.

EXAMPLE 1: CUSTOMER CODE 0xD1E2

The Customer ID code “0xD1E2” should be stored in the ID locations 200-203 like this:

200: 0000 0000 1101
201: 0000 0000 0001
202: 0000 0000 1110
203: 0000 0000 0010

Reading these four memory locations, even with the code protection bit programmed would still output on Port A the bit sequence “1101”, “0001”, “1110”, “0010” which is “0xD1E2”.

Note: *Microchip will assign a unique pattern number for QTP and SQTP requests and for ROM devices. This pattern number will be unique and traceable to the submitted code.*

2. CONFIGURATION WORD

The configuration word is the very first memory location which is accessed after entering the program/verify mode of the GMS77C1000/1001. It contains the two bits for the selection of the oscillator type, the watchdog timer enable bit, and the code protection bit. All other bits (4 through 11) are read as ’1’s.

One-Time-Programmable (OTP) devices may have the oscillator configuration bits “FOSC0” and “FOSC1” set by the factory and are tested accordingly. Therefore, it is essential that the inputs RA0 and RA1 are held at '1's when programming the “WDTE” and/or the “CP” bit of the configuration word. Otherwise, the factory tested and selected oscillator configuration could be overwritten and the functionality of the device is not guaranteed any more.

FIGURE 2-1: CONFIGURATION WORD FOR GMS77C1000/1001

-	-	-	-	-	-	-	-	CP	WDTE	FOSC1	FOSC0	Register: CONFIG Address: FFFh
bit11	10	9	8	7	6	5	4	3	2	1	bit0	

bit 11-4: **Unimplemented**: Read as ‘0’

bit 3: **CP:** Code protection bits
1 = Code protection off
0 = Code protection on

bit 2: **WDTE:** Watchdog timer enable bit
1 = WDT enabled
0 = WDT disabled

bit 1-0: **FOSC1:FOSC0:** Oscillator selection bits
11 = RC oscillator
10 = HS oscillator
01 = XT oscillator
00 = LP oscillator

3. CODE PROTECTION

The program code written into the EPROM can be protected by writing to the "CP" bit of the configuration word. All memory locations starting at 0x40 and above are protected against programming. It is still possible to program locations 0x00 through 0x3F, the ID locations, and the configuration word.

Note: *Locations [0x000 : 0x03F] are not secure after code protection.*

3.1 Programming Locations 0x000 to 0x03F after Code Protection

In code protected parts, the contents of the program memory cannot be read out in a way that the program code can be reconstructed. A location when read out will read as: 0000 0000 xxxx where xxxx is the XOR of the three nibbles.
For example, if the memory location contains 0xC04 (movlw 4), after code protection the output will be 0x008.
In addition, all memory locations starting at 0x40 and above are protected against programming. It is still possible to program locations 0x000 through 0x03F and the configuration word. However, performing a verify with activated code protection logic puts a 4-bit wide "checksum" on PORTA while the 8-bits of PORTB are read as '0's. The checksum is computed as follows:

The four high order bits of an instruction word are "XOR'ed" with the four middle and the four low order bits, and the result is transferred to PORTA. All memory locations are affected.
To program location 0x000 to 0x03F in a code protected part, the programmer should program one nibble at a time and verify the result through the XOR'ed output. For example, to program a location with 0xA93, first program the location with 0xFF3, verify checksum to be 0x003; then program the location with 0xF93 and verify the XOR'ed output to be 0x00C and finally program the location with 0xA93 and verify the read-out to be 0x006.

3.2 Embedding Configuration Word and ID Information in the Hex File

To allow portability of code, a GMS77C1000/1001 programmer is required to read the configuration word and ID locations from the hex file when loading the hex file. If configuration word information was not present in the hex file then a simple warning message may be issued. Similarly, while saving a hex file, all configuration word and ID information must be included. Configuration word should have the address of 0xFFF. ID locations are mapped at addresses described in Section 1.6.1 and Table 3-1. An option to not include this information may be provided.
Hynix feels strongly that this feature is important for the benefit of the end customer.

Table 3-1 : Configuration Word

* GMS77C1000 (CP enable pattern: 000000000XXX)

Program Memory Segment	R/W in Protected Mode	R/W in Unprotected Mode
Configuration Word (0xFFF)	Read Scrambled, Write Enabled	Read Unscrambled, Write Enabled
ID Words [0x200 : 0x203]	Read Scrambled, Write Enabled	Read Unscrambled, Write Enabled
[0x040 : 0x1FF]	Read Scrambled, Write Disabled	Read Unscrambled, Write Enabled
[0x000 : 0x03F]	Read Scrambled, Write Enabled	Read Unscrambled, Write Enabled

* GMS77C1001 (CP enable pattern: XXXXXXXX0XXX)

Program Memory Segment	R/W in Protected Mode	R/W in Unprotected Mode
Configuration Word (0xFFF)	Read Scrambled, Write Enabled	Read Unscrambled, Write Enabled
ID Words [0x400 : 0x403]	Read Scrambled, Write Enabled	Read Unscrambled, Write Enabled
[0x040 : 0x3FF]	Read Scrambled, Write Disabled	Read Unscrambled, Write Enabled
[0x000 : 0x03F]	Read Scrambled, Write Enabled	Read Unscrambled, Write Enabled

3.3 Checksum

3.3.1 CHECKSUM CALCULATIONS

Checksum is calculated by reading the contents of the GMS77C1000/1001 memory locations and adding up the opcodes up to the maximum user addressable location, e.g., 0x1FF for the GMS77C1000. Any carry bits exceeding 16-bits are neglected. Finally, the configuration word (appropriately masked) is added to the checksum. Checksum computation for each member of the GMS77C1000/1001 devices is shown in Table .

The checksum is calculated by summing the following:

- The contents of all program memory locations
- The configuration word, appropriately masked
- Masked ID locations (when applicable)

The least significant 16 bits of this sum is the check-sum.

The following table describes how to calculate the checksum for each device. Note that the checksum calculation differs depending on the code protect setting. Since the program memory locations read out differently depending on the code protect setting, the table describes how to manipulate the actual program memory values to simulate the values that would be read from a protected device. When calculating a checksum by reading a device, the entire program memory can simply be read and summed. The configuration word and ID locations can always be read.

Note that some older devices have an additional value added in the checksum. This is to maintain compatibility with older device programmer checksums.

Table 3-2 : CHECKSUM COMPUTATION

Device	Code Protect	Checksum*	Blank Value	0x723 at 0 and max address
GMS77C1000	OFF ON	SUM[0x000:0x1FF] + CFGW & 0x00F + 0x0FF0 SUM_XOR4[0x000:0x1FF] + CFGW & 0x00F	0x0DFF 0x1E07	0xFC47 0x1DF5
GMS77C1001	OFF ON	SUM[0x000:0x3FF] + CFGW & 0x00F + 0x0FF0 SUM_XOR4[0x000:0x3FF] + CFGW & 0x00F	0x0BFF 0x3C07	0xFA47 0x3BF5

Legend:

CFGW = Configuration Word

SUM[a:b] = Sum of locations a through b inclusive

SUM_XOR4[a:b] = XOR of the four high order bits with the four middle and the four low of memory location order bits summed over the locations a through b inclusive. For example, location_a = 0x123 and location_b = 0x456, then SUM_XOR [location_a : location_b] = 0x0007.

SUM_ID = ID locations masked by 0xF then made into a 16-bit value with ID0 as the most significant nibble. For example, ID0 = 0x1, ID1 = 0x2, ID3 = 0x3, ID4 = 0x4, then SUM_ID = 0x1234.

*Checksum = Sum of all individual expressions **modulo** [0xFFFF]

+ = Addition

& = Bitwise AND

4. PROGRAM/VERIFY MODE ELECTRICAL CHARACTERISTICS

4.1 DC Program Characteristics)

.

Table 4-1 : DC CHARACTERISTICS (TA = +10°C TO +40°C) (25°C IS RECOMMENDED

Parameter No.	Symbol	Characteristics	Min.	Typ.	Max.	Units	Conditions
PD1	V_{DDP}	Supply voltage during programming	4.75	5.0	5.25	V	Note 1
PD2	I_{DDP}	Supply Current (from V_{DD})			25.0	mA	V_{DD} = 5.0V, Fosc1 = 5MHz
PD3	V_{DDV}	Supply Voltage during verify	V_{DD}min		V_{DD}max		
PD5	$V_{HH}2$	Supply Voltage during verify	11.5		12.5	V	
PD6	I_{HH}	Voltage on $\overline{\text{RESET}}$ during programming			100	mA	
PD7	$I_{HH}2$	Current into $\overline{\text{RESET}}$ pin during programming (EC0=0)		10.0	25.0	mA	V_{HH} = 12.5V, V_{DD} = 5.0V
PD8	V_{IL}	Input Low Voltage	V_{SS}		0.15V_{DD}	V	
PD9	V_{IH}	Input High Voltage	0.85V_{DD}	5.0	V_{DD}	V	

Note 1: Device must be verified at minimum and maximum operating voltages specified in the data sheet.

4.2 AC Program and Test Mode Characteristics

Table 4-1 : AC CHARACTERISTICS (TA = +10°C TO +40°C, V_{DD} = 5.0V ± 5%) (25°C IS RECOMMENDED)

Parameter No.	Symbol	Characteristics	Min.	Typ.	Ma.x	Units	Conditions
P1	T_R	$\overline{RESET}$ Rise Time	0.15	1.0	8	µs	
P2	T_F	$\overline{RESET}$ Fall Time	0.5	2.0	8	µs	
P3	T_{PS}	Program Mode Setup Time	1.0			µs	
P4	T_{ACC}	Data Access Time			250	ns	
P5	T_{DS}	Data Setup Time	1.0			µs	
P6	T_{DH}	Data Hold Time	1.0			µs	
P7	T_{OE}	Output Enable Time	0		100	ns	
P8	T_{OZ}	Output Disable Time	0		100	ns	
P9	T_{PW}	Programming Pulse Width	10.0	100		µs	
P10	T_{PWF}	Programming Pulse Width		10,000		µs	Configuration Word only
P11	T_{RC}	Recovery Time	10.0			µs	
P12	F_{OSC}	Frequency on Xin	DC		5	MHz	For incrementing of the PC

FIGURE 4-1: PROGRAMMING AND VERIFY TIMING WAVEFORM

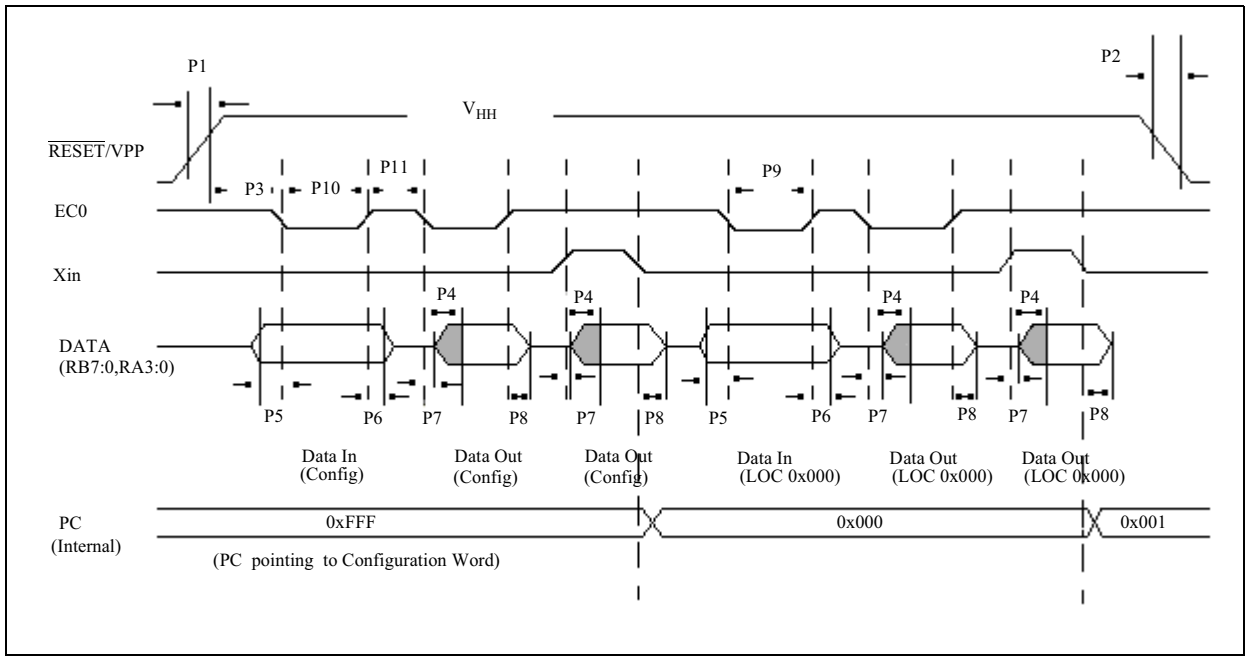
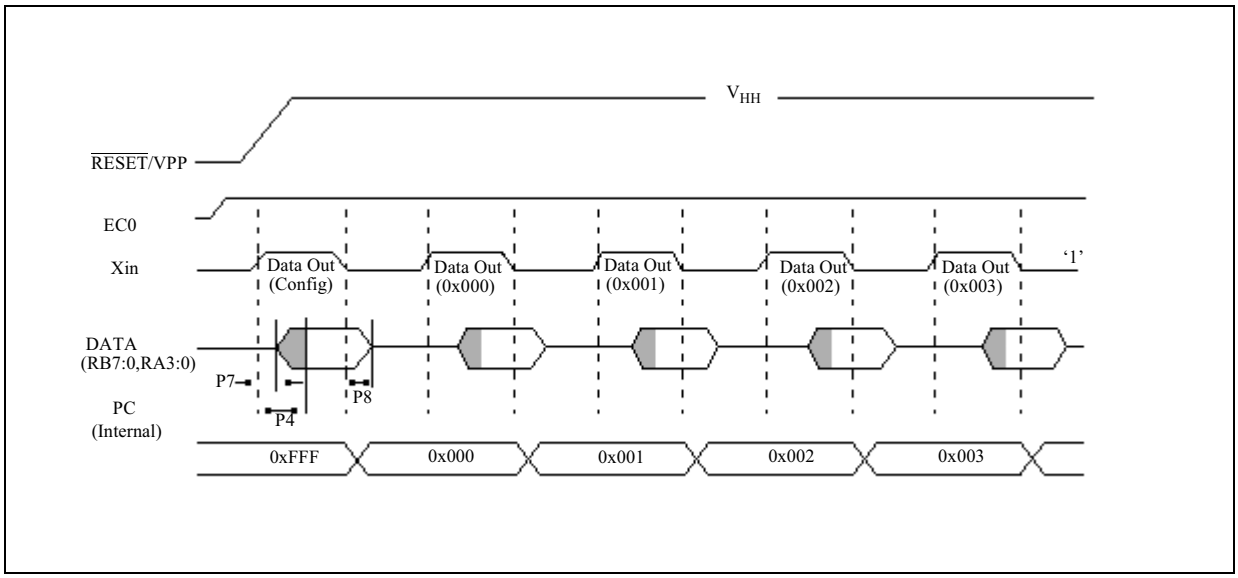


FIGURE 4-2: SPEED VERIFY WAVEFORM



Note: *Vpp Signal should br raised to high level straihtforwardly. Refer to following 2 waveform.*