8051 & 68HC908
In-System Programmer
Technical Manual

Date: 30 May 2002

Document Revision: 1.01

BiPOM Electronics

11246 S. Post Oak #205, Houston, Texas 77035
Telephone: (713) 661- 4214 Fax: (713) 661- 4201
E-mail: info@bipom.com
Web: www.bipom.com

WARRANTY:

BiPOM Electronics warrants In-System Programmer for a period of 1 year. If the chip becomes defective during this period, BiPOM will at its option, replace the chip. This warranty is voided if the product is subjected to physical abuse or operated outside stated electrical limits. BiPOM Electronics will not be responsible for damage to any external devices connected to In-System Programmer. BiPOM Electronics disclaims all warranties express or implied warranties of merchantability and fitness for a particular purpose. In no event shall BiPOM Electronics be liable for any indirect, special, incidental or consequential damages in connection with or arising from the use of this product. BiPOM Electronics' liability is limited to the purchase price of this product.

TABLE OF CONTENTS

1.Overview

Many modern Flash microcontrollers can be serially programmed while in the end application circuit. Customers can manufacture boards with un-programmed devices, and then program the microcontrollers just before shipping the product, allowing the most recent firmware or custom firmware to be programmed. This function of FLASH microcontrollers allows customers to create new program code, including a debugging also, in very easy manner. Downloading of new code to the microcontroller takes few seconds only. The microcontrollers from different manufacturers use various ways and methods of in-system programming. In-system programmer (PIC16C58B) gives universal standard RS-232 interface for programming all types of microcontrollers. Micro-IDE Integrated Development Environment from BiPOM Electronics fully supports in-system programming and debugging on the MINI-MAX/PRO-MAX boards using the serial port. Windows-based program WinLoad with graphical user interface is also provided to download programs to the boards, which contain in-system programmer. To build your own board you need to install in-system programmer by standard way that is used on BiPOM Single Board Computers. This includes whole software package from BiPOM Electronics to standard software tools of developers. There are two versions of in-system programmer:

- 8051 Loader PIC
- 68HC908 Loader PIC

8051 Loader PIC is designed for a support of Flash microcontrollers of MCS-51 family. Second programmer supports the MC68HC908GP32 32K Flash microcontroller from Motorola. Both the in-circuit programmers have also additional functions for driving of some peripheral devices on computer boards:

- 4-bit PWM (Pulse Width Modulator) output can be used to control external equipment/device. On BiPOM SBC's, this output is used for programmable contrast adjustment of LCD
- Programmable WDT (Watch Dog Timer). Host timeout range is 1..128 sec

Both the in-system programmers can work in two modes:

- Program Mode
- Run Mode

Program mode is a special mode when the in-system programmer communicates to host computer and writes/reads program code to/from Flash Memory of main microcontroller. Run mode is a main mode when the in-system programmer services additional functions.

In addition, the in-system programmer can write/read EEPROM AT24C65, that can be installed to the board, via I2C Bus by standard way, such as writing/reading of Flash memory of main microcontroller.

2. Serial Interface

The in-system programmer communicates over an asynchronous, UART interface at 19200 baud, no parity, 8 data bits, and 1 stop bit. All signals of in-system programmer should have CMOS voltage levels. Special converter, such as MAX232, should be used to convert the in-system programmer's RTS, RXD and TXD pins to/from RS232 levels. The converter has built-in voltage-doubler and inverter that generates +/- 10 Volts for RS232 logic levels. When a host computer or PC communicates with the in-system programmer, the host computer sends a request to the in-system programmer, and the in-system programmer returns a response. The host computer acts as a master, initiating all communications, and the in-system programmer acts as a slave, responding with a reply. In most cases the serial port is shared between in-system programmer and main microcontroller that is installed on the board. RTS line of PC solves a problem of common access. For successful communications, first the PC should switch the in-system programmer to Program mode. For this operation PC should make the RTS line of communication port as logical "0" (+12V should appear on the RTS pin of COM port). In-System programmer is checking a state of this signal on PORTA0 (pin 17 of PIC16C58) all the time. When RTS is low the in-system programmer will be switched to Program mode. If Program mode is activated by RTS line, the in-system programmer will make Reset State for main microcontroller. By this way the in-system programmer occupies UART interface. All in-system programmer commands and replies are wrapped in a proprietary Communications Protocol to insure the integrity and reliability of the data exchange. The Communications Protocol for the serial interface and the Command Set for the in-system programmer are described in detail in the sections that follow. Table 1 shows a wiring diagram of the in-system programmer's interface for communications.

Note that these signals are received/transmitted with a MAX232 CMOS/RS-232 line driver.

In-System Programmer PIC16C58	Pin	Pin	Host PC 9-pin male COM
PORTA1	18	3	Transmit Data (TXD)
POTRA2	1	2	Receive Data (RXD)
PORTA0	17	7	RTS

Table 1

Please refer to the schematics of single board computers for more details on RS-232 connections (see SCHEMATICS Section).

3. Communications Protocol

Each command to the in-system programmer consists of a request and a response. Each request to the in-system programmer consists of a command byte, any additional data required by the command and a check sum byte. Each response from the in-system programmer consists of a command byte, a status byte, the response data if any and a check sum byte. Both the command to the in-system programmer and the response from the in-system programmer are wrapped in a communications protocol layer.

Request: <command><additional data><check sum>

Response: <command><status><response data><check sum>

A verification check sum should be computed on all received messages from the in-system programmer and compared with the received check sum. If the verification check sum matches the received check sum the data from the in-system programmer was received correctly with a high degree of certainty.

CHECK SUM is calculated by adding all the bytes from a packet into a single byte. **STATUS** informs about a result of executed command.

4. In-System Programming (MCS-51)

The in-system programmer **8051 Loader PIC** supports several popular microcontrollers of MCS-51 family:

- AT89S8252, AT89S53
- P89C51RB2, P89C51RC2, P89C51RD2
- DS5000-8, DS5000-32, DS5000T-32T, DS89C420.

Micro-IDE Integrated Development Environment from BiPOM Electronics fully supports In-System Programming and Debugging using the serial port. There is also standalone loader from BiPOM Electronics (WinLoad) that supports In-System Programming.

4.1 AT89S53, AT89S8252.

AT89S53 micro-controller can be re-programmed remotely over the RS-232 interface using the in-system programmer (second microcontroller on the board, PIC16C58). The in-circuit programming feature simplifies program development on the board since downloading programs from a host PC takes only few seconds. User programs can also be debugged over the serial port. The on-chip Downloadable Flash of AT89Sxx allows the program memory to be reprogrammed in-system through an SPI serial interface. The in-system programmer is a bridge between RS232 and SPI interfaces. Host PC sends the necessary request (write, read or erase chip) through RS232-interface, the in-system programmer decodes this request and sends the necessary SPI request to AT89Sxx.

The in-system programmer can work in the two modes:

- Program Mode
- Run Mode

Run mode is a standard mode when AT89Sxx is running its own program. Program mode is a special mode when the reset pin of AT89Sxx is pull down. Holding RESET active forces the SPI bus into a slave input mode and allows the program memory to be written-to read-from.

For In-System Programming, first the PC should switch the board to Program mode. For this operation PC should make the RTS line of communication port as logical "0" (+12V should appear on the RTS pin of COM port). PIC16C58 is checking the state of this signal on pin 18 all the time. When RTS is low the PIC will turn on Program mode for Atmel chip. When RTS is high the PIC will turn off Program mode and A89Sxx will start a running of the program. The user can use this opportunity for resetting of the board. Then the PC should send the Set Type Request. After the PIC16C58 has received this request, it sends the Programming Enable Instruction to AT89Sxx via SPI. This instruction enables the SPI programming of flash memory. After this operation the PC can send any request (write, read, erase) to the board.

Other words, the following steps should be taken to read/write/erase the Flash memory of ATMEL microcontroller:

- Set Program mode
- Execute Set Type Command
- Execute Write/Read/Erase Commands
- Set Run mode

Table 2 shows a wiring diagram of the in-system programmer to ATMEL microcontroller (AT89S8252, AT89S53).

Note that ATMEL chip has PLCC-44 package.

In-System Programmer PIC16C58	Pin	Pin	ATMEL microcontroller
PORTB0	6	7	P1.5
POTRB1	7	8	P1.6
PORTB2	8	9	P1.7
PORTB3	9	10	RESET

Table 2

Please refer to the schematics of single board computers for more details on SPI connections (see 8051 SCHEMATIC).

CHECK_SUM is calculated by adding all the bytes from a packet into a single byte. **STATUS** informs about a result of executed command.

STATUS = 0 - OK
STATUS = 1 - CHECK SUM ERROR
STATUS = 2 - CHIP TYPE ERROR
STATUS = 3 - ENABLE PROGRAMMING ERROR
STATUS = 4 - WRITE BUFFER ERROR
STATUS = 5 - READ BUFFER ERROR

Set Type Command

Set Type Request

This request enables the SPI programming of flash memory. PC is sending this request to the in-system programmer at the first time.

< SET_TYPE_COMMAND = 1> <Type = 1><CHECK_SUM = 2>

Set Type Reply

The in-system programmer is sending this reply to the PC after the Set Type Request is received and enabling of SPI programming of ATMEL microcontroller was successful.

< SET_TYPE_COMMAND = 1> <STATUS = 0 ><CHECK_SUM = 1>

Note. If STATUS has no zero value it means the in-system programmer error.

Write Command

Write Request

This request allows to write the buffer of data bytes to the flash memory of ATMEL microcontroller. The maximum length of data buffer is 32 bytes.

< WRITE_BUFFER_COMMAND> <TYPE_MEMORY><LENGTH_BUFFER> <ADDRESS&0x00FF>< (ADDRESS >>8) &0x1F ><… DATA BYTES …><CHECK_SUM>

WRITE_BUFFER_COMMAND (e.g. 2) is the command to write the data buffer to the Flash memory of ATMEL chip.

TYPE_MEMORY defines the flash memory of ATMEL microcontroller.
TYPE_MEMORY = 1 for flash memory of AT89S53 (ADDRESS = 0..0x1FFF).
TYPE_MEMORY = 5 for flash memory of AT89S53 (ADDRESS = 0X2000…0x3000).
TYPE_MEMORY = 5 for EEPROM data of AT89S8252.

LENGTH_BUFFER defines a length of data buffer.

Write Reply

The in-system programmer is sending this reply to the PC after the Write Request is received and a writing of memory is successful.

< WRITE_BUFFER_COMMAND = 2> <STATUS = 0 ><CHECK_SUM = 2>

Note. If STATUS has no zero value it means the board error.

Read Command

Read Request

This request allows to read the buffer with data bytes from the flash memory of ATMEL microcontroller. The maximum length of data buffer is 32 bytes.

< READ_BUFFER_COMMAND> <TYPE_MEMORY><LENGTH_BUFFER> <ADDRESS&0x00FF>< (ADDRESS >>8) &0x1F ><CHECK_SUM>

READ_BUFFER_COMMAND (e.g. 3) is the command to read the data buffer from the flash memory of ATMEL microcontroller.

TYPE_MEMORY defines the flash memory of ATMEL microcontroller.
TYPE_MEMORY = 1 for flash memory of AT89S53 (ADDRESS = 0..0x1FFF).
TYPE_MEMORY = 5 for flash memory of AT89S53 (ADDRESS = 0X2000…0x3000)
TYPE_MEMORY = 5 for EEPROM data of AT89S8252.
LENGTH_BUFFER defines a length of data buffer.

Read Reply

The in-system programmer is sending this reply to the PC after the Read Request is received and reading of microcontroller memory is successful.

<READ_BUFFER_COMMAND = 3><STATUS = 0 ><…DATA BYTES …><CHECK_SUM >

Note. If STATUS has no zero value it means the board error.

Erase Command

Erase Chip Request
This request erases all the Flash memory of ATMEL microcontroller.

< ERASE_CHIP_COMMAND = 4> <CHECK_SUM = 4>

Erase Chip Reply

The in-system programmer is sending this reply to the PC after the Erase Chip Request is received and erasing of AT89Sxx is successful.

< ERASE_CHIP_COMMAND = 4> <STATUS = 0 ><CHECK_SUM = 4>

Note. If STATUS has no zero value it means the board error.

4.2 DS5000-8, DS5000-32, DS5000T-32, DS89C420.

When a used main microcontroller on the board is the DS5000 micro-controller from Dallas Semiconductor (BiPOM Electronics Part#: DS5000-32-12 or DS5000T-32-12), the board can be programmed without removing the micro-controller from its socket. Programs can be developed on a host PC, downloaded to board through the serial port and executed. DS5000-32 has 32 Kilobytes of battery-backed RAM that can be used as program memory, data memory or both. DS5000T-32 is the same as DS5000-32-12 with the addition of battery- backed real time clock. Microcontrollers from Dallas Semiconductor have serial bootstrap loader. If this type of a microcontroller is installed on the board the in-system programmer can enter this chip into special mode when the boot loader is running. The following steps should be taken to execute the boot loader for program downloading and running of user program on the target board:

- Set Program mode
- Send Set Type Request to the board
- Receive a reply about the status of monitor mode entering
- Download program using ROM boot loader
- Send 'H' character to the board when downloading is complete
- Set RUN mode

Table 3 shows a wiring diagram of the in-system programmer to DALLAS microcontroller (DS89C420).

Note that DALLAS chip has PLCC-44 package.

In-System Programmer PIC16C58	Pin	Pin	DS89C420
PORTB4	10	32	PSEN
PORTB3	9	10	RESET

Table 3

CHECK_SUM is calculated by adding all the bytes from a packet into a single byte. **STATUS** informs about a result of executed command.

STATUS = 0 - OK
STATUS = 1 - CHECK SUM ERROR
STATUS = 2 - CHIP TYPE ERROR

Set Type Command

Set Type Request for DS5000

This request enters DS5000 to special monitor mode when serial bootstrap loader is running. PC is sending this request to the in-system programmer.

< SET_TYPE_COMMAND = 1> <Type = 3><CHECK_SUM = 4>

Set Type Request for DS89C420

This request enters DS89C420 to special monitor mode when serial bootstrap loader is running. PC is sending this request to the in-system programmer.

< SET_TYPE_COMMAND = 1> <Type = 4><CHECK_SUM = 5>

Set Type Reply

The in-system programmer is sending this reply to the PC after the Set Type Request is received and entering into monitor mode is successful.

< SET_TYPE_COMMAND = 1> <STATUS = 0 ><CHECK_SUM = 1>

Note. If STATUS has no zero value it means the board error.
Note. The reply is the same for both microcontrollers.

4.3 P89C51RB2, P89C51RC2, P89C51RD2.

P89C51Rx2 microcontroller has factory ROM boot loader. If this micro-controller is installed on the board the in-system programmer can enter P89C51Rx2 into special mode when the boot loader is running. The following steps should be taken to execute the boot loader for program downloading and running of user program on the target board:

- Set Program mode
- Send Set Type Request to the board
- Receive a reply about the status of monitor mode entering
- Download program using ROM boot loader
- Send 'R' character to the board
- Set RUN mode

Table 4 shows a wiring diagram of the in-system programmer to PHILIPS microcontroller (P89C51RX2). **Note that PHILIPS chip has PLCC-44 package.**

In-System Programmer PIC16C58	Pin	Pin	P89C51RX2
PORTB4	10	32	PSEN
PORTB3	9	10	RESET

Table 4

CHECK_SUM is calculated by adding all the bytes from a packet into a single byte. **STATUS** informs about a result of executed command.

STATUS = 0 - OK
STATUS = 1 - CHECK SUM ERROR
STATUS = 2 - CHIP TYPE ERROR

Set Type Command

Set Type Request

This request enters P89C51Rx2 to special monitor mode. PC is sending this request to the in-system programmer at the first time.

< SET_TYPE_COMMAND = 1> <Type = 2><CHECK_SUM = 3>

Set Type Reply

The in-system programmer is sending this reply to the PC after the Set Type Request is received and entering into monitor mode is successful.

< SET_TYPE_COMMAND = 1> <STATUS = 0 ><CHECK_SUM = 1>

5. In-System Programming (MC69HC908GP32)

MC68HC908GP32 micro-controller can be re-programmed remotely over the RS-232 interface using the in-system programmer on the board. The in-circuit programming feature simplifies program development since downloading programs from a host PC takes only few seconds.

Micro-IDE Integrated Development Environment from BiPOM Electronics fully supports In-System Programming of the MC68HC908GP32 using the serial port. MINI-MAX/908-C loader of Micro-IDE is very fast. First it downloads RAM resident loader into MC68HC908GP32 using the in-system programmer PIC16C58. When RAM resident loader is running the MINI-MAX/908-C loader can write/read/erase the board directly without PIC16C58. For example, MINI-MAX/908-C loader takes only 25 seconds for writing/reading of 32K Flash memory (a downloading of RAM resident loader takes 10 seconds additionally).

The on-chip Downloadable Flash memory of MC68HC908GP32 allows the program memory to be reprogrammed in-system through special pin (PTA0) dedicated to serial communication between ROM monitor and PIC16C58. PIC16C58 is a bridge between RS232 of PC host and serial interface of ROM monitor. PC sends requests (such as Write, Read or Erase Chip) through the RS232 interface, PIC16C58 decodes this request and sends the necessary request to MC68HC908GP32.

The in-system programmer can work in the two modes:

- Program Mode
- Run Mode

Run mode is a standard mode when MC68HC908GP32 is running its own program. Monitor mode is a special mode when the MC68HC908GP32 is running the ROM monitor. For In-System Programming, first the PC should switch the board to Monitor mode. For this operation PC should make the RTS line of communication port as logical "0" (+12V should appear on the RTS pin of COM port). PIC16C58 is checking the state of this signal on PORTA0 (pin17 of PIC16C58) all the time. When RTS is low the PIC will turn on Monitor mode for Motorola chip. When RTS is high the PIC will turn off Monitor mode and MC68HC908GP32 will start a running of the program. The user can use this opportunity for resetting of the board. Then, the PC should send the Set Type Request. After this operation the PC can send any request (such as Write, Read, Erase) to the board. Other words, the following steps should be taken to read/write/erase the Flash memory of MC68HC908GP32 microcontroller in case of using of ROM monitor:

- Set Program mode
- Execute Set Type Command
- Execute Write/Read/Erase/Read Stack Pointer/Run Commands
- Set Run mode

Executing of WRITE and READ commands takes too much time, because communications are very intensive. Each data byte should be transmitted twice through communication line of ROM monitor (PTA0 of MC68HC908GP32). ROM monitor echoes each data byte back to the in-system programmer. To speed up the download process the following algorithm can be used:

- Set Program mode
- Execute Set Type Command
- Use Write Command to download RAM resident loader
- Use Read Stack Pointer and Run commands to start RAM resident loader
- Set Run mode
- Communicate to RAM resident loader and execute all available commands of RAM resident loader (etc. Write/Read/Erase)

This algorithm is used when MINI-MAX/908-C loader of Micro-IDE downloads program code to flash memory of MC68HC908GP32.

Table 5 shows a wiring diagram of the in-system programmer to Motorola microcontroller (MC68HC908GP32).

Note that Motorola chip has QFP-44 package.

In-System Programmer PIC16C58	Pin	Pin	MC68HC908GP32
PORTB0	6	32	PTA0
POTRB1	7	39	PTA7
PORTB2	8	5	PTC3
PORTB6	9	3	PTC1
PORTB7	13	1	RESET

Table 5

Please refer to the schematics of single board computers for more details on ROM monitor connections (see 68HC908 SCHEMATIC).

Exiting monitor mode after it has been initiated by having a blank reset vector requires a power-on-reset (POR). Pulling RST low will not exit monitor mode in this situation.

CHECK_SUM is calculated by adding all the bytes from a packet into a single byte. **STATUS** informs about a result of executed command.

STATUS = 0 - OK
STATUS = 1 - CHECK SUM ERROR
STATUS = 2 - CHIP TYPE ERROR
STATUS = 3 - ENABLE PROGRAMMING ERROR
STATUS = 4 - WRITE BUFFER ERROR
STATUS = 5 - READ BUFFER ERROR
STATUS = 6 – SECURITY ERROR
STATUS = 7 – ECHO ERROR
STATUS = 8 – FRAME ERROR
STATUS = 8 – POWER UP ERROR

Set Type Command

Set Type Request

This request allows to check a status of ROM monitor entering. PC is sending this request to the board at the first time.

< SET_TYPE_COMMAND = 1> <Type = 3><CHECK_SUM = 4>

Set Type Reply

The in-system programmer is sending this reply to the PC after the Set Type Request is received and entering of MC68HC908GP32 to monitor mode is successful.

< SET_TYPE_COMMAND = 1> <STATUS = 0 ><CHECK_SUM = 1>

Note. If STATUS has no zero value it means the board error.

Write Command

Set Type Request

This request allows to write the buffer of data bytes to the memory. The maximum length of data buffer is 32 bytes.

< WRITE_BUFFER_COMMAND> <TYPE_MEMORY><LENGTH_BUFFER>
<ADDRESS&0x00FF>< (ADDRESS >>8) &0x1F ><… DATA BYTES
…><CHECK_SUM>

WRITE_BUFFER_COMMAND (e.g. 2) is the command to write the data buffer to the microcontroller.

TYPE_MEMORY defines the type memory of Motorola microcontroller.
TYPE_MEMORY = 1 for RAM memory of MC68HC908GP32.
TYPE_MEMORY = 2 for FLASH memory of MC68HC908GP32.
LENGTH_BUFFER defines a length of data buffer.

Write Reply

The in-system programmer is sending this reply to the PC after the Write Request is received and writing of memory is successful.

< WRITE_BUFFER_COMMAND = 2> <STATUS = 0 ><CHECK_SUM = 2>

Note. If STATUS has no zero value it means the board error.

Read Command

Read Request.

This request allows to read the buffer with data bytes from the memory. The maximum length of data buffer is 32 bytes.

< READ_BUFFER_COMMAND> <TYPE_MEMORY><LENGTH_BUFFER>
<ADDRESS&0x00FF>< (ADDRESS >>8) &0x1F ><CHECK_SUM>

READ_BUFFER_COMMAND (e.g. 3) is the command to read the data buffer from the board.

TYPE_MEMORY defines the type memory of Motorola microcontroller.
TYPE_MEMORY = 1 for RAM memory of MC68HC908GP32.
TYPE_MEMORY = 2 for FLASH memory of MC68HC908GP32.

LENGTH_BUFFER defines a length of data buffer.

Read Reply

The in-system programmer is sending this reply to the PC after the Read Request is received and reading of memory is successful.

<READ_BUFFER_COMMAND = 3><STATUS = 0 ><…DATA BYTES …><CHECK_SUM >

Note. If STATUS has no zero value it means the board error.

Erase Command

Erase Chip Request

This request erases all the flash memory of MC68HC908GP32.

< ERASE_CHIP_COMMAND = 4>

Erase Chip Reply

The in-system programmer is sending this reply to the PC after the Erase Chip Request is received and erasing of MC68HC908GP32 is successful.

< ERASE_CHIP_COMMAND = 4> <STATUS = 0 ><CHECK_SUM = 4>

Note. If STATUS has no zero value it means the board error.

Run Command

Run Request

This request allows to start the user program on MC68HC908GP32 using RUN command of ROM monitor. The RUN command tells the Motorola microcontroller to execute PULH and RTI instructions. Before sending the RUN command the host should modify the stacked CPU registers to prepare to run the user program. More information on RUN command can be obtained from Monitor ROM section of MC68HC908GP32 datasheet on www.motorola.com.

< RUN_COMMAND = 6>

Run Reply

The in-system programmer is sending this reply to the PC after Run Request is received and executing of Run command is successful.

< RUN_COMMAND = 6> <STATUS = 0 ><CHECK_SUM = 6>

Note. If STATUS has no zero value it means the board error.

Read Stack Pointer Command

Read Stack Pointer Request

This request allows to read a value of current stack pointer + 1 of MC68HC908GP32 using ReadSP command of ROM monitor.

< READSP_COMMAND = 7>

Read Stack Pointer Reply

The in-system programmer is sending this reply to the PC after Read Stack Pointer Request is received and executing of ReadSP command is successful.

<READSP_COMMAND=7><STATUS=0><MSB_SP><LSB_SP><CHECK_SUM >

Note. If STATUS has no zero value it means the board error.

READSP_COMMAND (e.g. 7) is the command to read the stack pointer from the Motorola microcontroller.
MSB_SP is most significant 8 bits of Stack Pointer.
LSB_SP is least significant 8 bits of Stack Pointer.
StackPointer = (MSB_SP<<8) | LSB_SP

6. In-System Programming (EEPROM)

Both the in-system programmers (**8051 Loader PIC** and **68HC908 Loader PIC**) support SERIAL EEPROM AT24C65 that can be installed to the board.

Micro-IDE Integrated Development Environment from BiPOM Electronics fully supports In-System Programming using the serial port. There is also stand-alone loader from BiPOM Electronics (WinLoad) that supports In-System Programming.

The EEPROM AT24C65 allows the data memory to be reprogrammed in-system through an I2C serial interface. The in-system programmer is a bridge between RS232 and I2C interfaces. PC sends the necessary request (write or read memory) through RS232 interface, the in-system programmer decodes this request and sends the necessary I2C request to AT24C65.

Table 6 shows a wiring diagram of the in-system programmer to ATMEL EEPROM (AT24C65).

In-System Programmer PIC16C58	Pin	Pin	AT24C65
PORTB1	7	6	SCL
PORTB2	8	5	SDA

Table 6

CHECK_SUM is calculated by adding all the bytes from a packet into a single byte. **STATUS** informs about a result of executed command.

STATUS = 0 - OK
STATUS = 1 - CHECK SUM ERROR
STATUS = 2 - CHIP TYPE ERROR
STATUS = 4 - WRITE BUFFER ERROR
STATUS = 5 - READ BUFFER ERROR

Write Command

Write Request

This request allows to write the buffer of data bytes to the data memory of EEPROM AT24C65. The maximum length of data buffer is 32 bytes.

< WRITE_BUFFER_COMMAND> <TYPE_MEMORY><LENGTH_BUFFER> <ADDRESS&0x00FF>< (ADDRESS >>8) &0x1F ><… DATA BYTES …><CHECK_SUM>

WRITE_BUFFER_COMMAND (e.g. 2) is the command to write the data buffer to the data memory of EEPROM AT24C65.

TYPE_MEMORY = 3 defines the data memory of EEPROM AT24C65.

LENGTH_BUFFER defines a length of data buffer.

Write Reply

The in-system programmer is sending this reply to the PC after the Write Request is received and a writing of EEPROM memory is successful.

< WRITE_BUFFER_COMMAND = 2> <STATUS = 0 ><CHECK_SUM = 2>

Note. If STATUS has no zero value it means the board error.

Read Command

Read Request

This request allows to read the buffer with data bytes from the data memory of EEPROM AT24C65. The maximum length of data buffer is 32 bytes.

< READ_BUFFER_COMMAND> <TYPE_MEMORY><LENGTH_BUFFER> <ADDRESS&0x00FF>< (ADDRESS >>8) &0x1F ><CHECK_SUM>

READ_BUFFER_COMMAND (e.g. 3) is the command to read the data buffer from the data memory of EEPROM AT24C65.

TYPE_MEMORY = 3 defines the data memory of EEPROM AT24C65.
LENGTH_BUFFER defines a length of data buffer.

Read Reply

The in-system programmer is sending this reply to the PC after the Read Request is received and reading of EEPROM memory is successful.

<READ_BUFFER_COMMAND = 3><STATUS = 0 ><…DATA BYTES …><CHECK_SUM >

Note. If STATUS has no zero value it means the board error.

7. Watchdog Timer

8051 Loader PIC has two Watchdog Timers:

- HARD WDT
- SOFT WDT

For now **68HC908 Loader PIC has** HARD WDT only.

HARD WDT is built-in WDT of the PIC16C58. The Watchdog Timer is a free running on-chip RC oscillator, which does not require any external components. This RC oscillator is separate from the RC oscillator of the OSC1/CLKIN pin. That means that the WDT will run even if the clock on the OSC1/CLKIN and OSC2/CLKOUT pins of the device has been stopped. During normal operation a WDT time-out generates a device RESET when PORTB0 input (pin 6 of PIC16C58) has low voltage level (logical “0”). This will cause a RESET of in-system programmer. Then, the in-system programmer will reset the main micro-controller. C-code below demonstrates how user application can use this feature.

Reset.zip contains this Micro-IDE project.

```
/*****************************************************************************
 Description: this example demonstrates how to do the full restart of
 MM51C board. When P1.5 line, which is connected to PORTB0, has low level
the WDT of PIC16C58 will restart all the system.

 This program is written using Micro-C Compiler from Dunfield Development
Systems.
*****************************************************************************/
#include <8051io.h>
#include <8051bit.h>                    /* Bit set/clear macros */
#include <8051reg.h>

#define WDTLINE                         P1.5

main()
{
/* Configure serial port */
        serinit(9600);
        printf ("\n\n**   START   **");
        clrbit(WDTLINE);
        printf ("\nWAIT  NEW START");
        for(;;); // Loop forever

}
```

SOFT WDT is programmable Watchdog timer. Host timeout range is 1-128 sec. Between time period the main microcontroller must send SET WDT command to the in-system programmer through I2C bus, otherwise the in-system programmer will reset all system. Table 2 shows a wiring diagram of the in-system programmer to ATMEL microcontroller (AT89S8252, AT89S53). This wiring allows to communicate to the in-system programmer through I2C bus. PORTB3 is used for driving of RESET pin of main microcontroller.

Note that ATMEL chip has PLCC-44 package.

In-System Programmer PIC16C58	Pin	Pin	ATMEL microcontroller
PORTB0	6	7	P1.5 (Slave Select)
POTRB1	7	8	P1.6 (SCL)
PORTB2	8	9	P1.7 (SDA)
PORTB3	9	10	RESET

Table 2

Please refer to the schematics of single board computers for more details on I2C connections (see 8051 SCHEMATIC).

If SET WDT command contains ZERO value as parameter, this disables SOFT WDT. The MSB (most significant bit) of SET WDT command should be “1” always. PORTB0 input must have logical “0” state during I2C communications to the in-system programmer. For example, the following sequence disables SOFT WDT

- PORTB0 input to ground
- Start condition on I2C bus
- Send address byte of the in-system programmer on I2C bus (0XB0)
- Send ZERO value of SET WDT. The transmitting byte should be calculated as 0x00 + 0x80
- Stop condition on I2C bus
- Apply logical “1” to PORTB0 input

Wdt.zip contains the Micro-IDE project that illustrates a using of SOFT WDT in more detail.

SOFT WDT is disabled on POWER ON.

8. PWM OUTPUT

Both the in-system programmers (**8051 Loader PIC** and **68HC908 Loader PIC**) have PWM output (PORTA3, pin 2 of PIC16C58). The PWM output operates in 4-bit mode. It means 16 voltage levels are available in case of using of this output as Digital-To-Analog Converter (DAC). Building of DAC is very comfortable for driving of LCD contrast. Only two passive external electronic components (resistor, 1K + capacitor, 10uF) are necessary to build this circuit.

Please refer to the schematics of single board computers for more details on DAC implementation (see 8051 SCHEMATIC).

Pwm.zip contains the Micro-IDE project that illustrates a using of this feature in more detail.

9. Version

There is special command that allows to read the serial number of firmware from Loader PIC. Response is an ASCII string of printable characters, for example “1.10”. The response consists 4 printable characters always.

Version Command

Version Request

< VERSION_COMMAND = 5>

Version Reply

The in-system programmer is sending this reply to the PC after Version Request is received.

< VERSION_COMMAND = 7> <4 CHARS ><CHECK_SUM>

10. Crystal Oscillator

Both the in-system programmers (**8051 Loader PIC** and **68HC908 Loader PIC**) operate in HS oscillator mode. In HS mode, a crystal is connected to the OSC1/CLKIN and OSC2/CLKOUT pins to establish oscillation (Figure 1). The PIC16C58 oscillator design requires the use of a parallel cut crystal (see also 8051 SCHEMATIC).

Resonator frequency is different for the in-system programmers:

- XTAL = 8 MHz for 8051 Loader PIC
- XTAL = 9.8304 MHz for 68HC908 Loader PIC

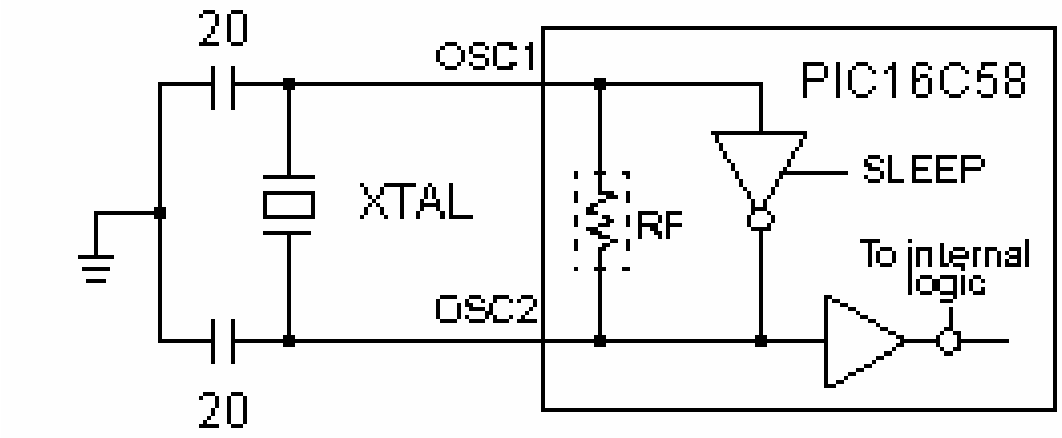
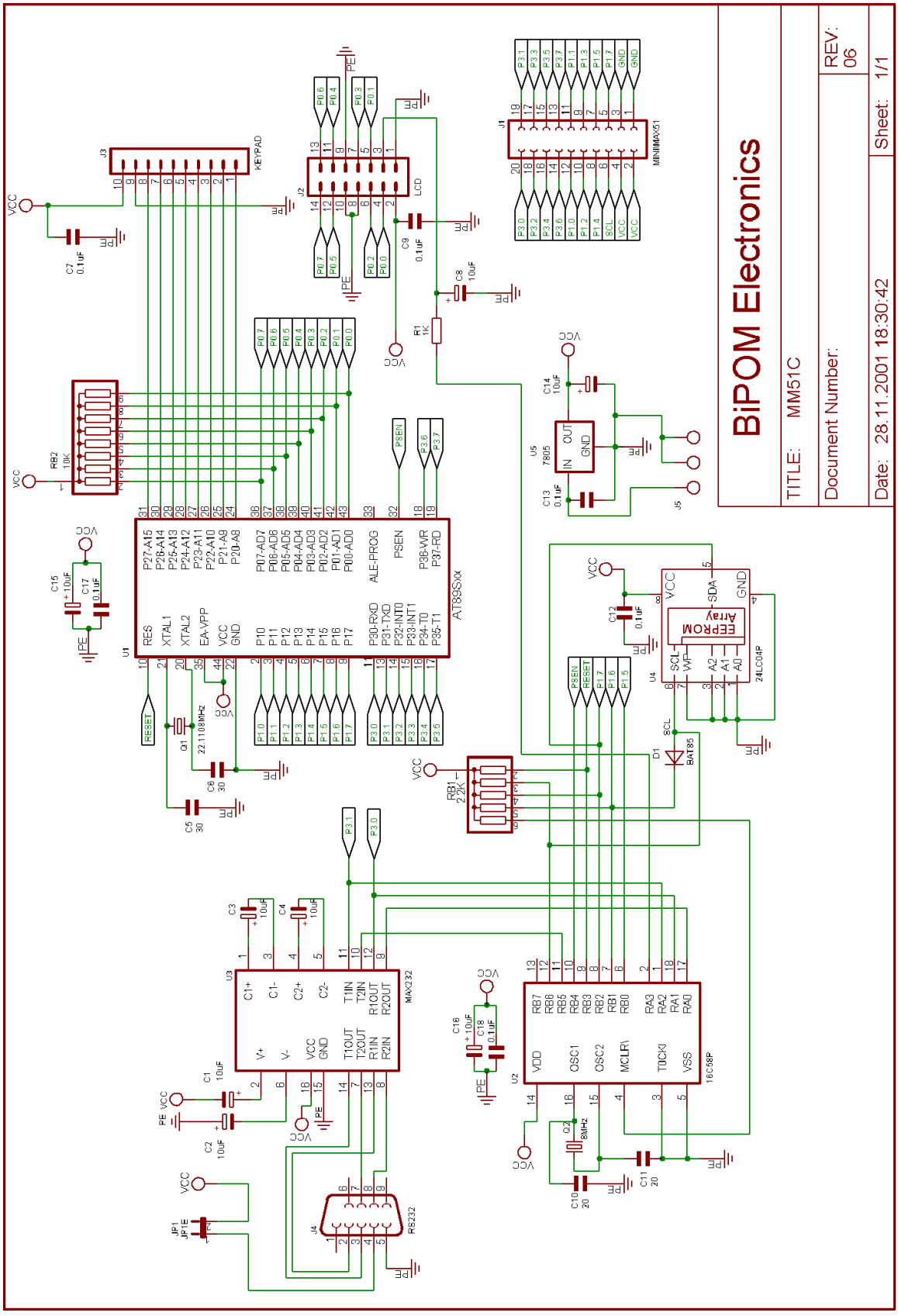


Figure 1

External oscillator design can be used also. This design is very suitable for the board based on using of MC68HC908GP32 microcontroller.

Please refer to the schematics of single board computers for more details on crystal oscillator design (see 68HC908 SCHEMATIC).

11. 8051 Schematic



12. 68HC908 Schematic