Installing and Using the Keil Monitor-51 — Application Note 152

May 31, 2000, Munich, Germany

by Keil Support, Keil Elektronik GmbH support.intl@keil.com ++49 89 456040-0

This Application Note describes the steps that are required to install and use the Keil Monitor-51 on a user specific hardware. The Keil Monitor-51 allows you to connect your 8051 hardware to the µVision2 Debugger. You can use the powerful debugging interface to test application programs in your target hardware.

For further information about using the Keil Monitor-51 together with the µVision2 Debugger refer to the User's Guide ***Getting Started and Creating Applications with C51*** *(KEIL\C51\HLP\GS51.PDF)*, ***Chapter 11. Using Monitor-51***.

Hardware and Software Requirements

The following requirements must be met for Monitor-51 to operate correctly:

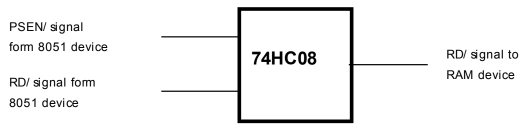
- 8051 CPU or derivative
- 5 Kbyte external code memory (EPROM) starting at address 0 (loaded with Monitor-51 software)
- 256 Byte external data memory (XDATA RAM) and 5 Kbytes trace buffer (optional). Additionally, the external data memory must be big enough to hold the complete application (code and data). **All** these external data memory areas must be *von Neumann* wired, this means that access is possible from XDATA and CODE space. A common way to do this is to connect the CPU signals /PSEN and /RD to a AND gate. The output of this AND gate is then connected to the /RD pin of the RAM.
- serial interface with a timer as baudrate generator.
- Between 1 and 5 port pins if you are using a banked hardware (for 2 – 32 banks). See example hardware schematics in the *8051 Utilities User's Guide under "Bank Switching Configuration"* for more details. All these memory banks have to be *von Neumann* wired!
- additional 6 bytes stack space (IDATA) in the user program to be tested.

All other hardware components can be used by the application.

Von-Neumann wired code/xdata Memory

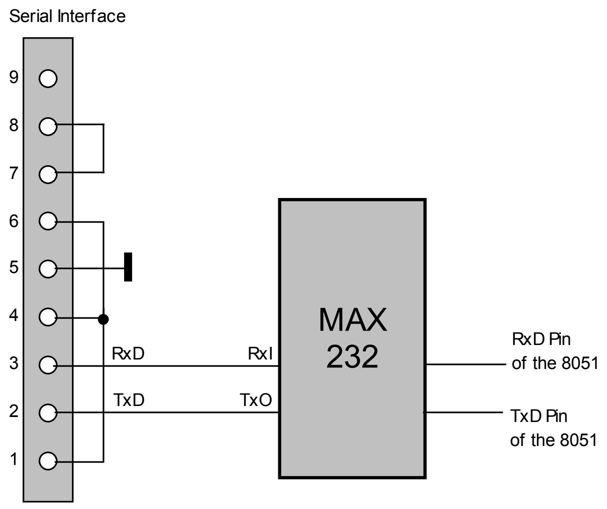
The Monitor requires that the program you are debugging is located in RAM space. For setting breakpoints in your code, the Monitor modifies the user code and inserts ACALL instructions at all breakpoint locations. Therefore you need to configure your code memory as von-Neumann memory.

Von-Neumann means that you can read physically the same memory bytes from code and xdata space. This is necessary to download software into **code** space since the 8051 does not provide CPU instructions to write into code memory. Typically a AND gate is used to combine the RD/ and PSEN/ signals of the CPU and generate a RD/ signal for the RAM device as shown in the figure on the left.



Serial Interface

Monitor-51 works with any standard serial interface and requires only the signals `TRANSMIT DATA`, `RECEIVE DATA` and `SIGNAL GROUND` from the RS232 or V.24 line. However, in most cases, some additional connections are required by the PC COM interface, to enable transmit and receive data. If you are using a 9-PIN standard connector on your application board you should therefore connect the pin 7 to pin 8, and pin 1 to pin 4 and pin 6. A typical schematic using an MAX 232 interface device is shown on the left.



Hardware with Boot Logic

Schematic will be added later.

Monitor-51 Configuration

The Monitor-51 can be adapted to different hardware configurations using the INSTALL batch file in the folder \KEIL\C51\MON51. This utility is invoked from a DOS command prompt and has the following command line syntax:

```
INSTALL  serialtype  [xdatastart  [codestart  [BANK][PROMCHECK]]]
```

The parameters of **INSTALL.BAT** are explained in the following.

serialtype defines the I/O routines used for the serial interface as explained in the table below.

Serial Type	Serial Interface	Clock Source	Baud Rate	CPU Clock	Processors
0	0	Timer 1	9600 bps	11,059 MHz	all 8051 variants
1	0	Int. Baudrate generator	9600 bps	12,000 MHz	80515(A), 80517(A)

Serial Type	Serial Interface	Clock Source	Baud Rate	CPU Clock	Processors
2	0	Timer 2	9600 bps	12,000 MHz	8052 and compatibles
3	1	Int. Baudrate generator	9600 bps	12,000 MHz	80517(A)
4	0	Timer 2	9600 bps	12,000 MHz	Dallas 80C320 /520/530
5	1	Timer 2	9600 bps	12,000 MHz	Dallas 80C320 /520/530
6	ext. UART 16450/16550	Ext. Crystal 3,686400 MHz	9600 bps	don't care	All
7	0	Timer 1	Self adjusting	don't care	All
8	0	Timer 2	Self adjusting	don't care	8052 and compatibles
9	0	Int. Baudrate generator	Self adjusting	don't care	80515A, C505C C515C,80517(A)
10	1	Int. Baudrate generator	Self adjusting	don't care	80517(A)
11	0	Timer2	Self adjusting	don't care	Dallas 80C320 /520/530
12	1	Timer 2	Self adjusting	don't care	Dallas 80C320 /520/530

xdatastart specifies the page number of the xdata memory area used by Monitor-51. The argument is a HEX value between **0** and **FF**. The default value is FF. Example: when ***xdatastart*** is **FF**, the memory area from X:0xFF00 to X:0xFFFF is used by Monitor-51 for internal variables and cannot be used by the user application. This memory area needs to be von-Neumann RAM that can be accessed from code and xdata space.

codestart specifies the page number of the code memory area for the Monitor-51 program code. The Monitor code requires typically 4 … 5 KBytes. The argument is a HEX value between 0 and **F0**. The default value is 0.

The option **BANK** creates a Monitor-51 version for a code banked target system. The file MON_BANK.A51 defines the hardware configuration of the banking hardware. See section below for further information about hardware configurations with code banking.

If the Monitor is created with the option **PROMCHECK**, the Monitor-51 checks on CPU reset if an EPROM or a RAM is present at code address 0. If an EPROM is detected, a JMP 0 instruction is executed that starts the code in the EPROM. **PROMCHECK** should be specified if the Monitor-51 code remains in the target system after the application has been programmed into an EPROM.

Example

```
INSTALL  8  7F  0
```

Creates a Monitor-51 version with self-adjusting baudrate that uses Timer 2 as the baudrate generator. The xdata space for internal Monitor-51 variables is between X:0x7F00 .. X:0x7FFF. The Monitor-51 code starts at address C:0x0000. This batch file creates the file MON51.HEX that can be burn into an EPROM.

NOTE
The file \KEIL\C51\MON51\MON51.PDF contains detailed information about the Monitor-51 configuration files and hardware requirements.

Configuration Files Settings

Depending on the target hardware and the memory areas used by Monitor-51, some settings in the files INSTALL.A51 and MON_BANK.A51 need to be modified.

Settings in INSTALL.A51

Interrupt offset:

When the Monitor-51 code is installed at ROM address 0, an application would never be able to reach an interrupt vector. Therefore, Monitor-51 diverts all interrupt vectors to a higher (RAM) address, where the application's interrupt vectors can be loaded. Interrupt offset has no meaning when the Monitor-51 code is not located at address 0.
Example: The code of Monitor-51 is installed at address 0. The 'v.Neumann' wired RAM starts at 8000H. The interrupt offset has to be set to 8000H and the application has to be linked to 8000H and above.

```
INT_ADR_OFF EQU 8000H   ; INTERRUPT VECTOR OFFSET IF MONITOR
                        ; IS INSTALLED AT ADDRESS 0000H
```

Changing the baudrate of a serial interface:

Depending on the oscillator frequency of the target system and the baudrate which should be used, it might be necessary to change the reload value of a baudrate timer. Therefore, search for the label 'InitSerial:' in the code section which initializes the serial interface to be used.

Example: Timer 2 of a 8052 should be used as baudrate generator (install option: serial = 2) . The 8052 is clocked with 12 MHz and the baudrate should be 4800 bps. Therefore, the RCAP2L register has to be initialized to the value 0B2H instead of 0D9H. Please refer to the manual of the corresponding 8051 derivative to see how the baudrate is calculated.

```
$IF (SERIAL = 2)
;*********************************************************************
;*  Using TIMER 2 to Generate Baud Rates (only for 8052)             *
;*  Oscillator frequency = 12.000 MHz                                *
;*  Set Baudrate to 9600 Baud                                        *
;*********************************************************************
RCAP2L  DATA    0CAH
RCAP2H  DATA    0CBH
T2CON   DATA    0C8H
 InitSerial:     PROMCHECK               ; Check if PROM in System
                MOV     T2CON,#34H
                MOV     RCAP2H,#0FFH
                MOV     RCAP2L,#0D9H
                MOV     SCON,#01011010B ; Init Serial Interface
                JMP     Mon51
$ENDIF
```

Other Functions and Definitions in INSTALL.A51

Since most serial interface configurations are already predefined, it is extremely seldom that there is a need to modify the communication functions listed below. Please note, that it is not allowed to modify any other register value (e.g. DPTR) without saving and restoring it.

INITSERIAL: This function initializes the serial interface and maybe other peripherals. At the end of this function a jump to MON51 instead of a RET is necessary.

Ser_Int_Adr:	This definition specifies the interrupt vector address of the serial interface. The default value is 023H for a standard 8051, but it can be set to any other interrupt vector address (e.g. external interrupt). If the serial interface is not able to generate an interrupt, set this value to any unused interrupt vector address.
INSTAT:	This function returns the receive status of the serial interface in the C-flag. C = 1 means character received; C = 0 means no character received.
OUTSTAT:	This function returns the transmit status of the serial interface in the C-flag. C = 1 means character transmitted; C = 0 means character not yet transmitted.
INCHAR:	This function returns the received character in the accumulator.
OUTCHAR:	This function transmits the character in the accumulator to the serial interface.
CLR_TI:	This function clears the transmit interrupt flag of the serial interface
SET_TI:	This function sets the transmit interrupt flag of the serial interface.
CLR_RI:	This function clears the receive interrupt flag of the serial interface.
CLR_SER_IE:	This function clears the interrupt enable flag of the serial interface.
SET_SER_IE:	This function sets the interrupt enable flag of the serial interface.
BEFORE_GO:	This function is called before a application is started with GO or P-step. It is empty by default (only RET). This function can be used to generate a signal at a port pin or to change the memory configuration.
AFTER_GO:	This function is called when a application has been stopped with a breakpoint. It is empty by default (only RET). It should do the opposite of BEFORE_GO.
WR_CODE:	Since it is not possible to write to the code memory of a 8051, the external RAM needs to be 'von Neumann' wired. Therefore, write access to the code memory is usually done with a MOVX @DPTR,A instruction. However, if the target system allows a different way to do this (e.g. different address area) it can be adapted here.

Bank Switching Configuration

When you install Monitor-51 for code banking, you must specify the number of code banks your hardware provides as well as how the code banks are switched. This is done by changing constants that are defined in the assembly module MON_BANK.A51.

MON_BANK.A51 Constants

The banking method as well as the number of banks and thus the number of address lines used are configured using this source file. MON_BANK.A51 contains EQU statements at the beginning which are used for the configuration. Following is a listing of these as well as a description of each.

```
;************************ Configuration Section *******************************
?B_NBANKS       EQU     8         ; Define max. Number of Code Banks (not      *
;                                 ; including XDATA or COMMON bank).           *
;                                 ; The max. value for ?B_BANKS is 32          *
;                                 ; possible values are: 1,2,3,...32           *
;                                                                              *
?B_MODE         EQU     0         ; 0 for Bank-Switching via 8051 Port         *
;                                 ; 1 for Bank-Switching via XDATA Port        *
;                                                                              *
?B_BANKSTART    EQU     08000H    ; defines the start address of the code      *
;                                 ; banking area                               *
;                                                                              *
?B_BANKEND      EQU     0FFFFH    ; defines the end address of the code        *
;                                 ; banking area                               *
;                                                                              *
?B_COMMON       EQU     0FFH      ; 0FFH if the COMMON area is not mapped into *
```

```
;                                ; a code bank.                          *
;                                ; otherwise ?B_COMMON must be set to the     *
;                                ; bank number which contains the COMMON area *
;                                                                        *
?B_XRAM          EQU     0FFH     ; 0FFH if the XDATA RAM area is not mapped   *
;                                ; into a code bank.                     *
;                                ; otherwise ?B_XRAM must be set to the bank  *
;                                ; number which contains the XDATA RAM area   *
;                                                                        *
?B_MON_DATA_BANK EQU    007H     ; Bank number where monitor data is stored   *
;                                                                        *
IF  ?B_MODE = 0;                                                         *
;------------------------------------------------------------------------*
; if ?BANK?MODE is 0 define the following values                         *
; For Bank-Switching via 8051 Port define Port Address / Bits            *
?B_PORT          EQU     P1       ; default is P6                        *
?B_FIRSTBIT      EQU     0        ; default is Bit 5                     *
;------------------------------------------------------------------------*
```

?B_NBANKS	indicates the number of banks to be supported. The number must be between 2 and 32. Only one 8051 address line (port terminal) is used for two banks. Three or four banks require two address lines. Five to eight banks require three address lines. Nine to sixteen banks require four address lines. Seventeen to thirty-two banks require five address lines.
?B_MODE	indicates if the bank switching code should use an 8051 port or an XDATA port for the address extension. A value of 0 defines an arbitrary 8051 port for the address extension. A value of 1 determines a XDATA port which is addressed in the external address space of the 8051.
?B_BANKSTART	specifies the start address of the banking area. The value 0 means, that there is no physical common area. In this case, the logical common area is automatically copied into every code bank during the load process of your application. The value 0FFFFH specifies, that you are running a banked version of Monitor-51 on a non-banked hardware. 8000H is defined as the default value.
?B_BANKEND	specifies the end address of the banking area. This value must be greater or equal than ?B_BANKSTART. 0FFFFH is defined as the default value.
?B_COMMON	indicates whether the physical common area must be accessed as a bank or not. The default value 0FFH specifies, that the common area is not mapped to a bank. Any other value specifies the bank number of the common area. This option is used to separate the common area and XDATA area, although the common area is 'von Neumann' mapped.
?B_XRAM	indicates whether the XDATA area must be accessed as a bank or not. The default value 0FFH specifies, that the XDATA area is not mapped to a bank. Any other value specifies the bank number of the XDATA area. This option is used to separate the common and XDATA area. Both, ?B_COMMON and ?B_XRAM have to be set to 0FFH or both to different values.
?B_MON_DATA_BANK	specifies the bank number of the XDATA area used by Monitor-51. In order to save memory space, the 256 byte monitor XDATA area and the optional 5KB trace memory can be located in a code bank instead of the common area.
?B_PORT	specifies the port address used to select the bank address. If the value 0 is used for ?B_MODE, ?B_PORT can be used to specify the address of the internal data port. In this case, the SFT address of an internal data port must be specified. P1 is defined as the default value for port 1.

?B_XDATAPORT	specifies the XDATA memory address used to select the bank address. If the value 1 is used for ?B_MODE, ?B_XDATAPORT defines the address of an external data port. In this case, an arbitrary XDATA address can be specified (address range 0H to 0FFFFH) under which a port can be addressed in the XDATA area. 0FFFFH is defined as the default value.
?B_FIRSTBIT	indicates which bit of the defined port is to be assigned first. The value ?B_FIRSTBIT EQU 3 (defined as the default when ?B_MODE is 0) indicates that P1.3 is to be used as the first port terminal for the address extension. If, for example two port terminals are used for the extension, P1.3 and P1.4 are used in this case. The remaining lines of the 8051 port can be used for other purposes. If the value 1 is selected for ?B_MODE, the remaining bits of the XDATA port cannot be used for other purposes.

Preparing Programs for Monitor-51

Compiling and linking programs for use with the 8051monitor on a target system requires a view additional steps depending on the Monitor-51 and hardware configuration.

Translating Modules

First, check out whether you can load your program at code address 0, or if the Monitor-51 is installed there. If your target board is equipped with a boot logic that allows to install the monitor at a higher address and the ‘von Neumann’ wired RAM is mapped to address 0, skip this section and go to ‘Linking/Locating Segments’ below.

If the monitor is installed at code address 0 and the ‘von Neumann’ wired RAM starts at a higher address, all interrupt vectors, the startup vector and all code segments must be relocated to this RAM start address. For the following examples, a RAM start address of 8000H is assumed.

Compiling C Modules

Translate your applications as you normally would, but additionally use the **INTVECTOR** directive to relocate all interrupt vectors to a higher address. For example, you may use the following command line.

```
C51 MYCODE.C INTVECTOR(0x8000)
```

When you are using µVision2, you can enter this offset under **Options for Target – C51 – Interrupt vectors at address**.

Translating Assembler Modules

Assembler modules may contain absolute segments (CSEG or ORG) and absolute references to CODE or XDATA locations. All these CODE segments and CODE references must be located in the ‘von Neumann’ wired RAM address range of your target board. Therefore, changes in your assembly source code may be necessary.
Relocatable CODE and XDATA segments (RSEG) are located by the linker L51/BL51. Options therefore are described later under ‘Linking/Locating Modules’.

Preparing STARTUP.A51

In order to locate the startup-vector to an address other than 0, it is necessary to modify the STARTUP.A51 file. Therefore, copy the STARTUP.A51 file from the 'C51\LIB\' directory into your project directory and search for the following line:

```
             CSEG  AT 0
?C_STARTUP:  LJMP  STARTUP1
```

This is the startup-vector definition. Change the address from 0 to the start address of your 'von Neumann' wired RAM.

```
             CSEG  AT 08000H
?C_STARTUP:  LJMP  STARTUP1
```

This STARTUP.A51 file must then be translated with A51 and linked to your application.
When you are using uVision, add the modified STARTUP.A51 file as a assembly source file to your project.

Preparing L51_BANK.A51

The module L51_BANK.A51 is only necessary when a code banked application should be generated. The constant settings in this file are similar to the settings in MON_BANK.A51. For more details, please refer to section 'Bank Switching Configuration' in chapter 1 of the 8051 utilities manual.
This L51_BANK.A51 file must then be translated with A51 and linked to your application.
When you are using µVision2, add the modified L51_BANK.A51 file as an assembly source file to your project.

Linking/Locating Modules

Locating CODE Segments

Again, it depends on the Monitor-51 configuration and the target hardware where the code and data segments have to be located. In any case, the code segments need to be located in the 'von Neumann' wired RAM area.

If Monitor-51 is installed at a high address and 'von Neumann' RAM is available at address 0, only the serial interrupt vector has to be reserved when it is used. Monitor-51 uses the serial interrupt vector only as an option. With the BL51 *CODE (0100H)* directive all relocatable code and constant segments are located at 100H and above, reserving all interrupt vectors.

```
BL51 MYCODE.OBJ, STARTUP.OBJ CODE (0100H)
```

When Monitor-51 is installed at address 0, the code segments have to be located where the 'von Neumann' RAM starts.

```
BL51 MYCODE.OBJ, STARTUP.OBJ CODE (08000H)
```

When you are using µVision2, you will find this option in the pull down menu 'Options' / 'BL51 Banked Linker' / 'Size/Location' / 'Code Address'.

Locating XDATA Segments

Since MONITOR-51 needs a ‘von Neumann’ wired memory architecture to load and debug code, the XDATA and CODE segments cannot be located at the same address. The application’s XDATA Segments have to be located either in the ‘von Neumann’ wired RAM above all CODE segments or in a separate XDATA area. Therefore, please check the linker map file (*.M51) of your application to figure out size and location of all code segments. With the BL51 XDATA directive all relocatable xdata segments are located at the specified address and above.

```
BL51 MYCODE.OBJ, STARTUP.OBJ CODE (0100H)  XDATA (04000H)
or
BL51 MYCODE.OBJ, STARTUP.OBJ CODE (08000H)  XDATA (0C000H)
```

Troubleshooting

If the Monitor-51 does not start correctly it is typically a problem of Monitor code and data locations or the initilization of the serial interface.

If the Monitor-51 stops working or behaves strange during debugging of your application, your application is most likely overwriting the user application program. This might happen when the user application makes xdata write accesses to the program code locations. Code and xdata memory must be non-overlapping areas, since the Monitor-51 requires *von Neumann* wired code space, which means that code and xdata space are physically the same memory area. You should therefore check the XDATA and CODE MEMORY MAPPING that is listed in the Linker MAP (*.M51) file and verify that code and xdata space are not overlapping.

If the Monitor-51 does not single step CPU instructions or if you cannot read or write SFR data locations the Monitor-51 data memory area cannot be accessed from code space. The Monitor-51 data memory must be also *von Neumann* wired xdata/code space.

During operation the Monitor might report the following errors:

Error Text	Description
CONNECTION TO TARGET SYSTEM LOST	µVision2 has lost the serial connection to the Monitor program. This error might occur because your program re-initializes the serial interface used by Monitor-51. This error also occurs when you single step in the serial I/O routines of your application.
NO CODE MEMORY AT ADDRESS xxxx	You try to download code into ROM space or non-existing memory. The code memory must be von Neumann wired xdata/code RAM.
CANNOT WRITE INTERRUPT VECTOR	The Monitor program cannot install the interrupt vectors for the Serial interface. This error occurs when the code memory at address 0 cannot be accessed. Most likely this space is not von Neumann wired.