NEC 78K0- Family On-Chip Emulation

Contents

1 Introduction

NEC 78K0 uses proprietary serial interface for in-circuit debugging of microcontrollers and FLASH programming. Such firmware is implemented on the CPU silicon providing a comprehensive set of debug functionalities. Communication interface uses CPU clock lines X1 and X2.

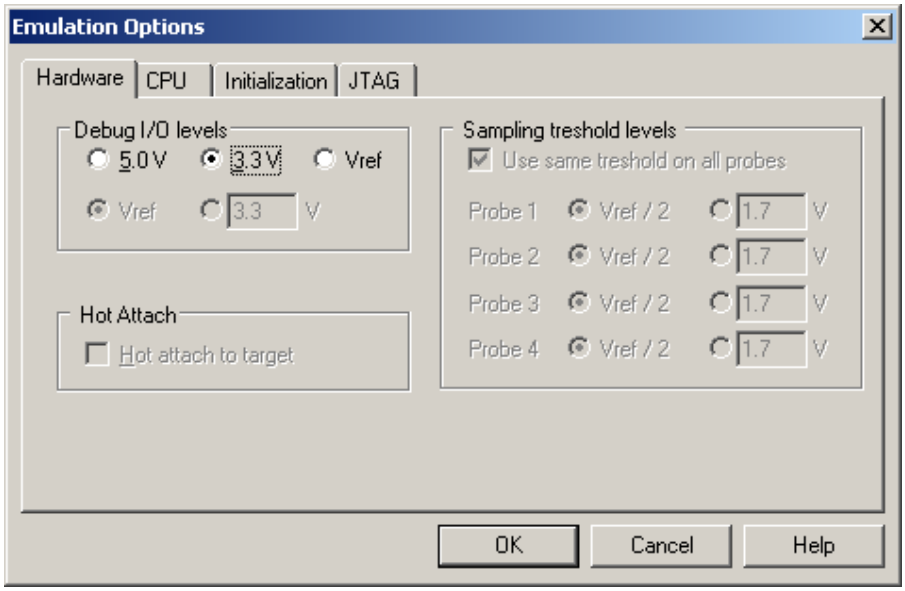
Debug Features

The 78K0 emulation system features:

- 1 execution hardware breakpoint
- Unlimited software breakpoints, including in the internal FLASH
- 1 access breakpoint inside the address range 0xF800-0xFFFF
- Internal FLASH programming

2 Emulation options

2.1 Hardware Options



Emulation options, Hardware pane

Debug I/O levels

The development system can be configured in a way that the debug signals are driven at 3.3V, 5V or target voltage level (Vref: 1.8V -5V).

When 'Vref' Debug I/O level is selected, a voltage applied to the belonging reference voltage pin on the target debug connector is used as a reference voltage for voltage follower, which powers buffers, driving the debug signals. The user must ensure that the target power supply is connected to the Vref pin on the target connector and that it is switched on before the debug session is started. If these two conditions are not meet, it is highly probably that the initial debug connection will fail already. However in some cases it may succeed but then the system will behave abnormal.

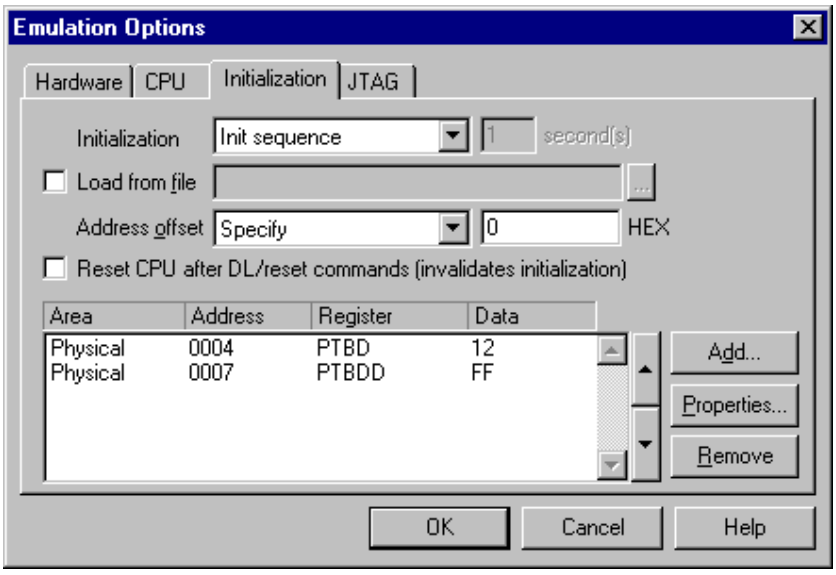
Initialization Sequence

Before the flash programming or download can take place, the user must ensure that the memory is accessible. This is very important since there are many applications using memory resources (e.g. external RAM, external flash), which are not accessible after the CPU reset. In that case, the debugger must execute after the CPU reset a so called initialization sequence, which configures necessary CPU chip selects and then the download or flash programming can actually take place. The user must set up the initialization sequence based on his application.

Note: Normally, there is no need for initialization sequence in case of a single chip application/CPU.

The initialization sequence can be set up in two ways:

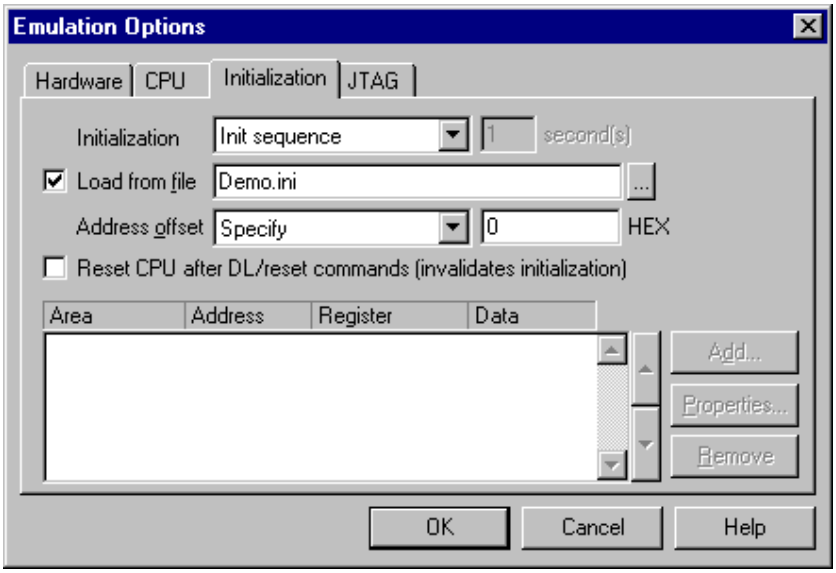
1. Set up the initialization sequence by adding necessary register writes directly in the Initialization page within winIDEA.



2. winIDEA accepts initialization sequence as a text file with .ini extension. The file must be written according to the syntax specified in the appendix in the hardware user's guide.

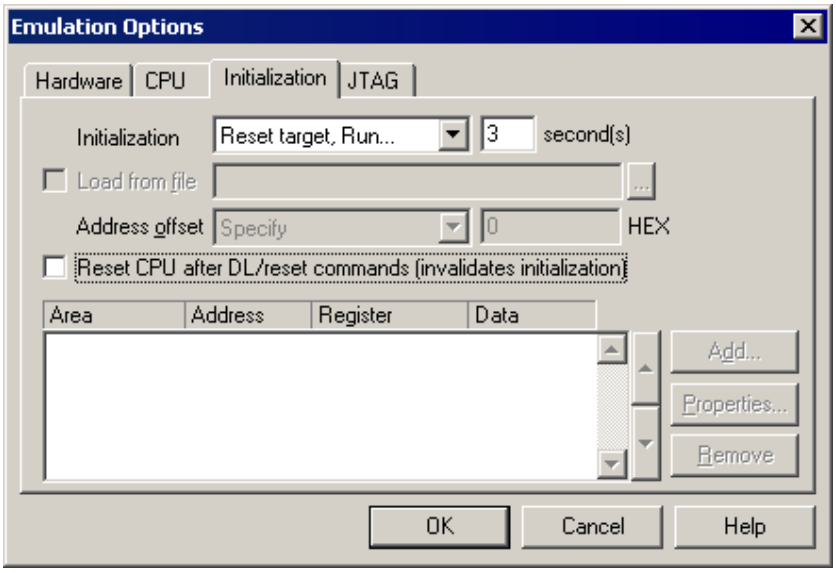
   Excerpt from the sample Demo.ini file:

```
S PTBD B 12              //comment
S PTBDD B FF
```



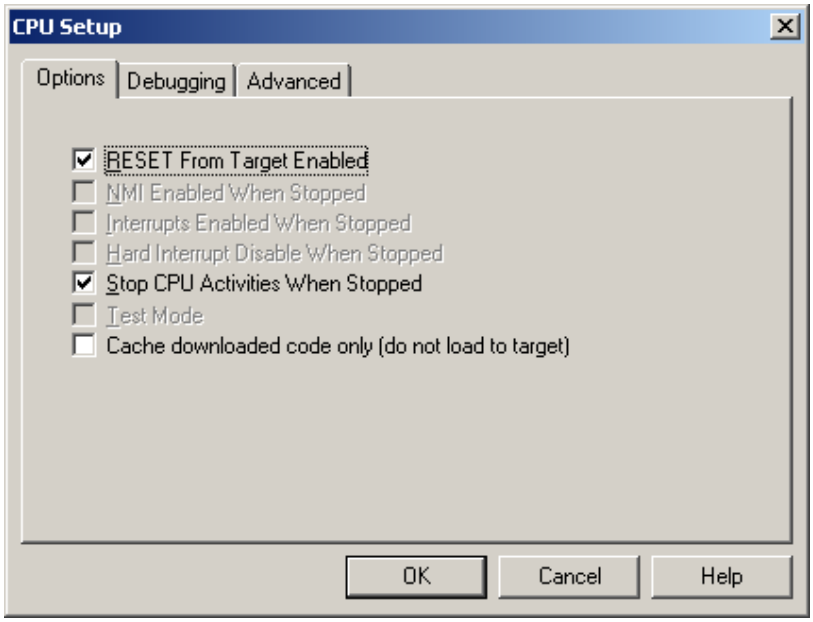
The advantage of the second method is that you can simply distribute your .ini file among different workspaces and users. Additionally, you can easily comment out some line while debugging the initialization sequence itself.

There is also a third method, which can be used too but it's not highly recommended for the start up. The user can initialize the CPU by executing part of the code in the target ROM for X seconds by using 'Reset and run for X sec' option.



3 CPU Setup

3.1 General Options



NEC 78k Family Advanced CPU Options

RESET From Target Enabled

Beside the debugger, the target can have additional external reset sources, like power-on reset, watchdog circuitry or even reset push-button. In general, it's recommended to disable all external reset sources in the target, which may disturb the debugger in a way that serial communication is lost and complete system needs to be reinitialized.

It's recommended that all reset sources are designed as an open drain type. 'Reset from Target Enabled' option in the 'CPU Setup/Options' tab must be normally checked to assure safer debugging. Then the debugger can detect any reset source and service it properly.

Since target reset lines are designed as an open drain type, the debugger can detect all resets, even if they have been initiated by hardware other than the emulator itself. In certain applications, though, the requirement to disable this type of checking is required.

To disable reset sources from the target to be detected by the debugger, uncheck the 'RESET From Target Enabled' option. In this case, only the emulator will be able to generate a reset and the debugger will ignore all reset sources from the target.

Note: Wrong setting of this option can significantly change the operation of the target!

Stop CPU Activities When Stopped

When the option is checked, all internal peripherals like timers and counters are stopped when the application is stopped. Otherwise, timers and counters remain running while the program is stopped. Usually, when the option is checked, the emulation system behaves more consistently while stepping through the program. While being aware of the consequences, it is up to the user whether the option is checked or not.

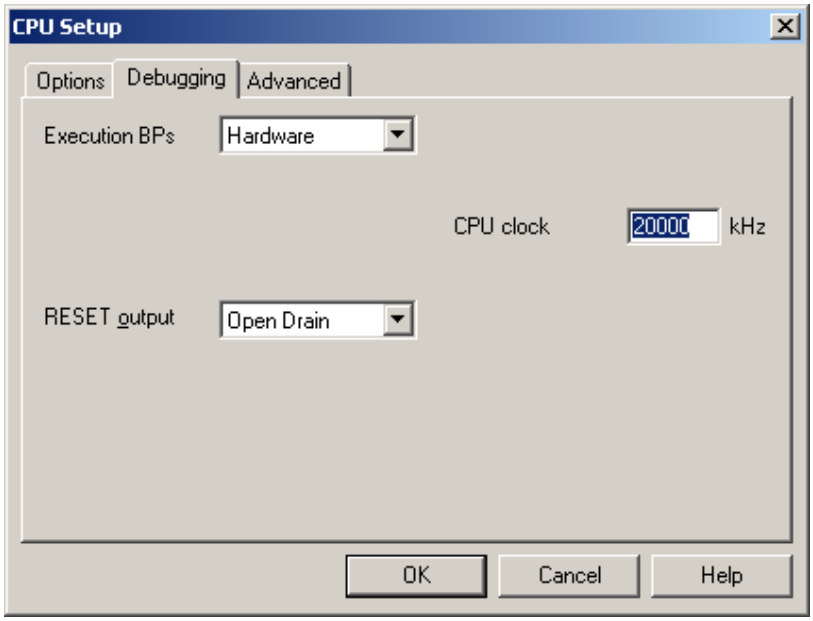
For instance, it's is recommend that a timer, which generates interrupts, is stopped when the application is stopped. Otherwise, the CPU would first service all pending interrupts (generated by the timer while the application was stopped) after the application is resumed. Such behaviour is far away from the actual behaviour of the target application.

Cache downloaded code only (do not load to target)

When this option is checked, the download files will not propagate to the target using standard debug download but the Target download files will.

In cases, where the application is previously programmed in the target or it's programmed through the flash programming dialog, the user may uncheck 'Load code' in the 'Properties' dialog when specifying the debug download file(s). By doing so, the debugger loads only the necessary debug information for high level debugging while it doesn't load any code. However, debug functionalities like ETM and Nexus trace will not work then since an exact code image of the executed code is required as a prerequisite for the correct trace program flow reconstruction. This applies also for the call stack on some CPU platforms. In such applications, 'Load code' option should remain checked and 'Cache downloaded code only (do not load to target)' option checked instead. This will yield in debug information and code image loaded to the debugger but no memory writes will propagate to the target, which otherwise normally load the code to the target.

3.2 Debugging Options



NEC 78k Family Debugging Options

Execution Breakpoints

Hardware Breakpoints

Hardware breakpoints are breakpoints that are already provided by the CPU. The number of hardware breakpoints is limited to one. The advantage is that they function anywhere in the CPU space, which is not the case for software breakpoints, which normally cannot be used in the FLASH memory, non-writeable memory (ROM) or self-modifying code. If the option 'Use hardware breakpoints' is selected, only hardware breakpoints are used for execution breakpoints.

Note that the debugger, when executing source step debug command, uses one breakpoint. Hence, when all available hardware breakpoints are used as execution breakpoints, the debugger may fail to execute debug step. The debugger offers 'Reserve one breakpoint for high-level debugging' option in the Debug/Debug

Options/Debugging' tab to circumvent this. By default this option is checked and the user can uncheck it anytime.

Software Breakpoints

Available hardware breakpoints often prove to be insufficient. Then the debugger can use unlimited software breakpoints to work around this limitation. Note that the debugger features unlimited software breakpoints in the NEC 78k CPU internal flash too.

When a software breakpoint is being used, the program first attempts to modify the source code by placing a break instruction into the code. If setting software breakpoint fails, a hardware breakpoint is used instead.

Using flash software breakpoints

A flash device has a limited number of programming cycles. Belonging flash sector is erased and programmed every time when a software breakpoint is set or removed. The debugger sets breakpoints hidden from the user also when a source step is executed. In worst case, a flash may become worn out due to intense and long lasting debugging using flash software breakpoints.

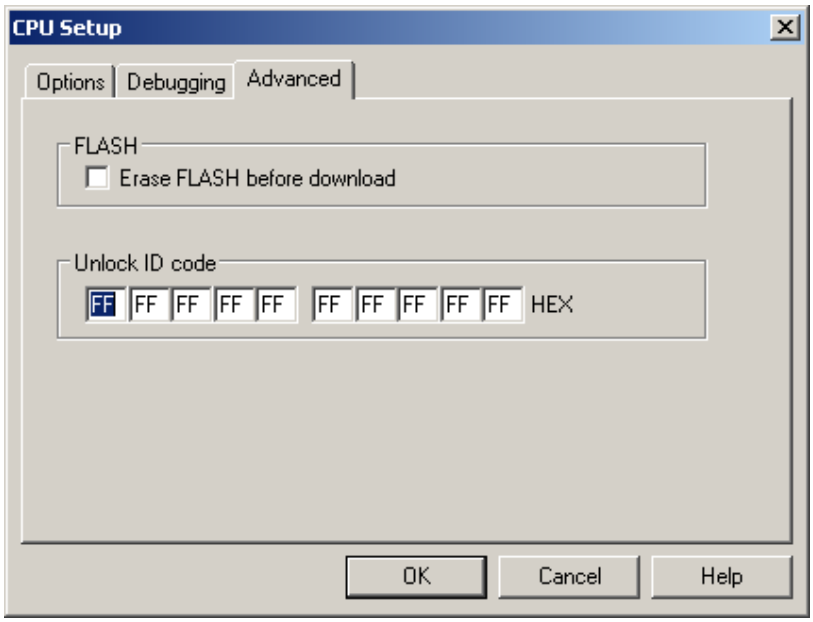
CPU Clock

Here you specify CPU clock frequency. This frequency is also used by serial debug interface and is also required by the internal flash programming.

RESET Output

Two types of RESET output, depending on the application used, are available – Open Drain or Push-Pull. The appropriate RESET output type can be selected here.

3.3 Advanced Options



NEC 78k Family Advanced CPU Options

Erase FLASH before download

Check the option if complete flash should be erased prior to the debug download. Otherwise, only the necessary sectors are erased.

Unlock ID code

After reset 10 bytes ID code specified in winIDEA is compared with 10 bytes from memory location (0x0085 – 0x008E). Based on ID check result and Monitor startup configuration byte (address 0x0084) emulation is enabled or not.

4 Internal FLASH Programming

NEC 78k CPUs have internal flash, which is programmed through standard debug download; thereby no standard FLASH setup dialog is available. The debugger recognizes which code from the download file fits in the FLASH. All necessary FLASH programming settings are done in the 'CPU Setup/Advanced' dialog.

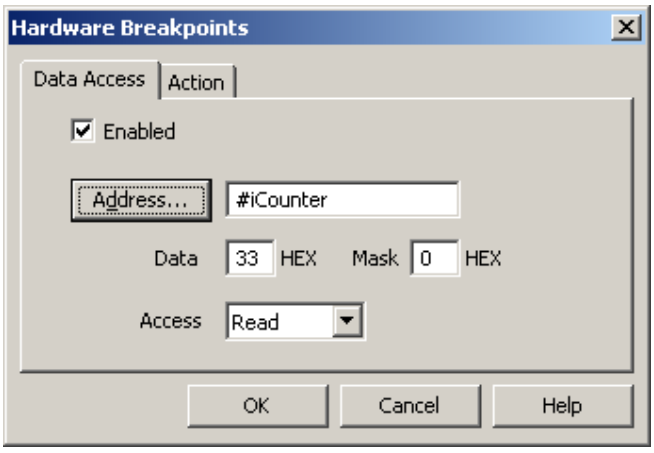
Internal FLASH is programmed using hidden monitor called SelfLIB.

5 Real-Time Memory Access

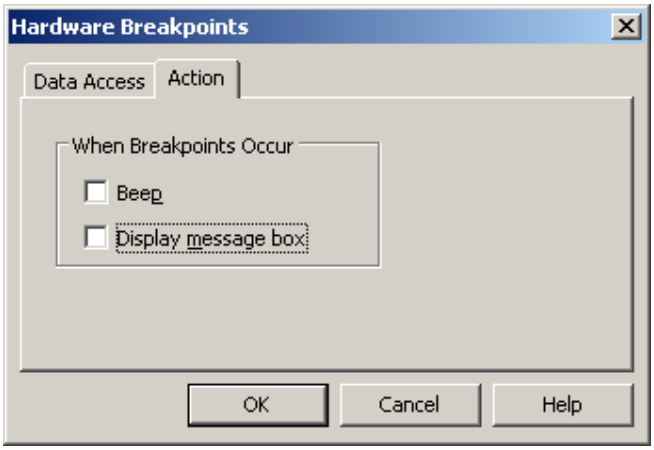
NEC 78k0 does not support real-time memory access.

6 Access Breakpoints

Access breakpoints are breakpoints that are already provided by the CPU. The number of access breakpoints is limited to one and can be set in a range 0xF800-0xFFFF. Data value, data mask and access type can be specified.



NEC 78k0 Hardware Breakpoint Configuration – Data Access tab



NEC 78k0 Hardware Breakpoint Configuration – Action tab

When access breakpoint is hit a display message can be shown or you can set a PC to play a sound.

7 Emulation Notes

Reserved memory locations for monitor are:

- 0x0002 - 0x0003: NMI vector
- 0x007E - 0x007F:CALLT [1F] vector
- 0x0084: Monitor startup configuration byte
- 0x008F – 0x01FF:Monitor

If you try to load user code or data to these reserved memory locations, winIDEA will display a warning message.

Additional memory locations which are used by monitor but are not reserved are:

- 0x0085 – 0x008E: 10 bytes ID code

8 Getting Started

1) Connect the system

2) Make sure that the target debug connector pinout matches with the one requested by a debug tool. If it doesn't, make some adaptation to comply with the standard connector otherwise the target or the debug tool may be damaged.

3) Power up the emulator and then power up the target.

4) Execute debug reset

5) The CPU should stop on location to which the reset vector points

6) Open memory window at internal CPU RAM location and check whether you are able to modify its content.

7) If you passed all 6 steps successfully, the debugger is operational and you may proceed to download the code in the internal CPU flash.

8) Check ‘Erase FLASH before download' options in the 'CPU Setup/Advanced' tab.

9) Specify the download in the 'Debug/Files for download/Download files' tab.

10) Execute Debug download, which should download the code in the internal CPU flash.

9 Troubleshooting

- Make sure that the power supply is applied to the target serial connector when ‘Vref’ is selected for Debug I/O levels in the Hardware/Emulator Options/Hardware tab, otherwise emulation fails or may behave unpredictably.

- When performing any kind of checksum, remove all software breakpoints since they may impact the checksum result.