

## NeuroMatrix® Software Development Kit (NM-SDK)

NM-SDK provides the essential tools for efficient, high-performance applications development. Now it's available on Win32 platforms.

NM-SDK includes:

- C++ compiler;
- Optimizing C++ compiler;
- Assembler;
- Linker;
- Object files librarian;
- Decoder of object and executable files;
- Multitarget JIT debugger;
- Instruction level simulator;
- Accurate cycle simulator;
- Set of system libraries.

The user can translate C/C++ programs into NM6403 assembly language source, translate assembly language programs into ELF object files with debugging information saved in DWARF format, make object libraries, create executable files for NM6403 with resolved references, debug executable files.

The Assembly language has intuitive syntax and is close to high level languages. It simplifies development and understanding of written codes.

### *Hardware and software requirements*

Minimum hardware and software requirements:

1. RAM 8 MB (minimum for Windows 95)
2. Windows 95 or above installed
3. at least 20 MB of free disk space

### *Interfaces*

All software tools except GUI Multitarget JIT debugger are Win32 console applications and have command line interfaces. Each tool has own user options described in respective user's documentation. Warnings and error messages have a unified format and can be output on console or into file.

### *C++ compiler*

The C++ Compiler is a full-featured compiler that translates C/C++ programs into NM6403 assembly language source. The following list describes key characteristics of the compiler:

- The C++ Compiler supports definition of C++ language described in preliminary standard ANSI X3J16/95-0029 with the exception some kind of templates.
- The compiler is designed as a Win32 console application and is controlled by command line options.
- The compiler package comes with the runtime library `libc.lib`. The source code of the contents of the library is available in the CRT directory of the installed NM-SDK. The library includes functions for time-keeping, dynamic memory allocation, data conversion, and floating-point arithmetic.

- The compiler supports a flat memory model. There are no any restrictions on object code size. Memory space is limited only by configuration of hardware.
- The compiler package includes a shell program, which enables the user to execute all steps of program translation into NM6403 executable code with one command.
- The compiler has straightforward calling conventions, allowing the user to easily write assembly and C/C++ functions that call each other.
- Executable and Linkable File (ELF) format allows the user to define system's memory map at a link time. This maximizes performance by enabling the user to link C/C++ code and data into specific memory areas.
- Debugging information format DWARF provides rich support for source-level debugging.

### *Optimizing C++ compiler*

The Optimizing C++ compiler is a full-featured compiler that translates C/C++ programs into NM6403 assembly language source. It is a new program in NM-SDK and can be used instead of the old C++ compiler. The Optimizing C++ compiler more close adherences to the C++ standard, including templates, and uses the enhanced optimizing algorithms.

### *Assembler*

The Assembler is software tool intended for translation of program source codes written in assembly language to object files of ELF format. Its main task is to construct a table of symbols and to conduct transformation of assembler lines into processor instructions.

The assembly language can contain macros and assembler directives. The assembler allows obtaining object code from assembly language program and as well listing of assembler program and listing of cross-references. The assembler permits to create macros libraries and expand them by adding new macros.

The assembler has a set of directives to store debug information in an assembler file. The assembler supports debug information representation according to the standard DWARF, version 2.0.

### *Linker*

When the linker processes input object files it fulfills the following function:

- combines sections with the same name and creates for them private relocation tables which are needed for tuning of references for specific configuration of the NM6403 memory;
- while building executable files with tuning for specific configuration of the NM6403 memory linker calculates symbols addresses and sections addresses, and adjust all references stored in the relocation tables;

- combines the sections to program segment to accelerate and simplify program loading to the NM6403 memory;
- resolves undefined external references among the input files;
- provides possibility of deleting unused sections and symbols and debugging information from the output file;
- delivers information on error found during linking process.

The linker supports different NM6403 memory configurations. For this purpose there has been developed C-like language. This language helps to describe:

- the ranges of accessible physical addresses and hardware characteristics of the NM6403 memory banks (physical memory configuration);
- the layout of the application program segments in the memory and their sizes (logic memory configuration);
- the layout of the program sections in the particular segments.

The description is contained in the special file called configuration file.

#### ***Object files librarian***

The object files librarian is intended for the work with object file libraries of ELF format. It fully supports this format with the exception of creating and processing dynamic libraries.

The librarian allows:

- to create libraries from the sets of object files,
- to list the content of libraries,
- to add, delete and replace files in libraries,
- to extract object files from libraries.

#### ***Decoder of object and executable files (dumper)***

The dumper is intended to decode the contents of object and executable ELF files. It allows the user to view the contents of object ELF files, object libraries and executable ELF files in the readable text format.

#### ***Software Simulators***

The software simulators are developed in 2 variants:

- an instruction level simulator;
- a cycle accurate simulator.

#### **Instruction level simulator**

The simulator is used to simulate the operation of NM6403. The simulator executes the user program code likewise NM6403 but it doesn't simulate run-time characteristics. A result of an execution any user program by simulator is the same as by NM6403.

#### **Cycle accurate simulator**

The cycle simulator is intended to simulate the operation of NM6403 and characterization the run-time of user programs. The cycle simulator executes the user program code likewise NM6403. Moreover it simulates run-time characteristics such as varies delays. However the memory access time is fixed, and equals to one cycle. A result of the execution any user program by cycle simulator is the same as by NM6403. A difference between the run time on simulator and NM6403 maybe exists only for a fragment of the user program, which depends on the pipeline and the memory access time. The cycle accurate simulator gives to the developer all data needed for the program optimization.

#### ***Multitarget JIT Debugger***

The Debugger is a windowing tool that enables NM program debugging on the local target from the Win32 host machine. The Debugger uses changeable debugging targets. A debugging target communicates with the host debugger through some type of universal processor interface to enable cross debugging. NM-SDK includes the following debugging targets:

- Instruction level NM6403 simulator
- NM6403 processor through link

Other debugging targets are provide with the boards support kits.

The Debugger provides the following services:

- controls program execution, using Instructions or Source windows,
- displays and modifies register value and memory contents,
- displays call stack and virtually unwinds the call stack,
- displays and modifies global objects of the program (function addresses and global variables),
- displays and modifies local variables of the frames,
- uses target specifics information which is provided by debugging target,
- sets conditional/unconditional breakpoints.

#### ***Set of system library***

NM-SDK includes:

- a subset of the standard run-time C library which does not include input/output and file system support functions;
- NM6403 timers control, interrupt processing and communication links control library;
- NM6403 vector operations library as a functional example.