

FUJITSU Software BS2000 POSIX-NSL

Version V10.0A
October 2016

Readme

All rights reserved, including intellectual property rights.
Technical data subject to modifications and delivery subject to availability. Any liability that the data and illustrations are complete, actual or correct is excluded. Designations may be trademarks and/or copyrights of the respective manufacturer, the use of which by third parties for their own purposes may infringe the rights of such owner.

© 2016 Fujitsu Technology Solutions GmbH

Fujitsu and the Fujitsu logo are trademarks or registered trademarks of Fujitsu Limited in Japan and other countries. BS2000 is a trademark of Fujitsu Technology Solutions GmbH in Germany and other countries.

1 Introduction	3
2 Product Components and Installation	4
2.1 BS2000 Installation	4
2.2 Installation to the POSIX File System	4
3 Functions	8
4 Restrictions	9

1 Introduction

This readme file contains notes concerning the software product POSIX-NSL which provides the NSL library for the development of POSIX software. The NSL-library contains the functions of the services TLI, XDR and RPC.

2 Product Components and Installation

2.1 BS2000 Installation

Following BS2000 files may be installed by IMON:

	MODE	SHARE	ACCESS	MIGRATE
\$<TSOS>.SYSLIB.POSIX-NSL.100	COM	Y	R	I
\$<TSOS>.SKMLIB.POSIX-NSL.100	X86	Y	R	I
\$<TSOS>.SYSRME.POSIX-NSL.100.D	COM	Y	R	I
\$<TSOS>.SYSRME.POSIX-NSL.100.E	COM	Y	R	I
\$<TSOS>.SYSSII.POSIX-NSL.100	COM	Y	R	A

- SYSLIB.POSIX-NSL.100 contains header files and modules in the /390 format for compilation and linking software by means of BS2000 and for the installation of the NSL library /usr/lib/libnsl.a to the POSIX file system.
- SKMLIB.POSIX-NSL.100 contains header files and modules in the X86 format for compilation and linking software by means of BS2000 and for the installation of the NSL library /usr/lib/libnsl.a respectively /usr/lib/X86/libnsl.a to the POSIX file system.

2.2 Installation to the POSIX File System

The installation to the POSIX file system has to be done using the POSIX installation program. (/START-POSIX-INSTALLATION, function Install Packages on POSIX). See manual "POSIX Basics" for more details.

The installation to the POSIX file system requires about 9 MB of temporary disk space in /tmp and about 9 MB of permanent disk space per each installed variant in /usr/lib or /usr/lib/X86.

The installation will take some time.

If the installation should not have been executed correctly, the text file /tmp/install.libnsl.err would contain notes on the error reasons.

Installed Libraries:

If only the BS2000 file SYSLIB.POSIX-NSL.100 is installed, only the library /usr/lib/libnsl.a with the /390 code variant will be created in the POSIX file system.

If several variants of the BS2000 libraries are installed, several variants of the NSL libraries will be created in the POSIX file system, too:

```
/usr/lib/libnsl.a          with /390 code format
/usr/lib/X86/libnsl.a     with X86 code format
```

In this case the linking with the X86 variant can be done by using the compiler option

```
-L /usr/lib/X86
```

Installed Files:

```
/usr/bin/rpcgen
/usr/bin/rpcgen_cpp

/usr/include/netconfig.h
/usr/include/netdb.h
/usr/include/netdir.h
/usr/include/tiuser.h
/usr/include/arpa.ftp.h
/usr/include/arpa/ftp.h
/usr/include/arpa/inet.h
/usr/include/arpa/inet.h
/usr/include/arpa.nameser.h
/usr/include/arpa.nameser.h
/usr/include/arpa.telnet.h
/usr/include/arpa/telnet.h
/usr/include/arpa.tftp.h
/usr/include/arpa/tftp.h
/usr/include/net.af.h
/usr/include/net/af.h
/usr/include/net.if.h
/usr/include/net/if.h
/usr/include/net.if_arp.h
/usr/include/net/if_arp.h
/usr/include/net.route.h
/usr/include/net/route.h
/usr/include/net.strioc.h
/usr/include/net/strioc.h
/usr/include/netinet.arp.h
/usr/include/netinet/arp.h
/usr/include/netinet.icmp_var.h
/usr/include/netinet/icmp_var.h
/usr/include/netinet.if_ether.h
/usr/include/netinet/if_ether.h
/usr/include/netinet.in.h
/usr/include/netinet/in.h
/usr/include/netinet.insrem.h
/usr/include/netinet/insrem.h
/usr/include/netinet.in_pcb.h
/usr/include/netinet/in_pcb.h
```

```
/usr/include/netinet.in_sysm.h
/usr/include/netinet/in_sysm.h
/usr/include/netinet.in_var.h
/usr/include/netinet/in_var.h
/usr/include/netinet.ip.h
/usr/include/netinet/ip.h
/usr/include/netinet.ip_icmp.h
/usr/include/netinet/ip_icmp.h
/usr/include/netinet.ip_str.h
/usr/include/netinet/ip_str.h
/usr/include/netinet.ip_var.h
/usr/include/netinet/ip_var.h
/usr/include/netinet.llcloop.h
/usr/include/netinet/llcloop.h
/usr/include/netinet.nihdr.h
/usr/include/netinet/nihdr.h
/usr/include/netinet.symredef.h
/usr/include/netinet/symredef.h
/usr/include/netinet.tcp.h
/usr/include/netinet/tcp.h
/usr/include/netinet.tcpi.h
/usr/include/netinet/tcpi.h
/usr/include/netinet.tcp_debug.h
/usr/include/netinet/tcp_debug.h
/usr/include/netinet.tcp_fsm.h
/usr/include/netinet/tcp_fsm.h
/usr/include/netinet.tcp_seq.h
/usr/include/netinet/tcp_seq.h
/usr/include/netinet.tcp_timer.h
/usr/include/netinet/tcp_timer.h
/usr/include/netinet.tcp_var.h
/usr/include/netinet/tcp_var.h
/usr/include/netinet.udp.h
/usr/include/netinet/udp.h
/usr/include/netinet.udp_var.h
/usr/include/netinet/udp_var.h
/usr/include/rpc.auth.h
/usr/include/rpc/auth.h
/usr/include/rpc.auth_des.h
/usr/include/rpc/auth_des.h
/usr/include/rpc.auth_sys.h
/usr/include/rpc/auth_sys.h
/usr/include/rpc.auth_unix.h
/usr/include/rpc/auth_unix.h
/usr/include/rpc.clnt.h
/usr/include/rpc/clnt.h
/usr/include/rpc.clnt_soc.h
/usr/include/rpc/clnt_soc.h
/usr/include/rpc.des_crypt.h
/usr/include/rpc/des_crypt.h
/usr/include/rpc.key_prot.h
/usr/include/rpc/key_prot.h
/usr/include/rpc.nettype.h
/usr/include/rpc/nettype.h
/usr/include/rpc.pmap_clnt.h
/usr/include/rpc/pmap_clnt.h
/usr/include/rpc.pmap_prot.h
/usr/include/rpc/pmap_prot.h
/usr/include/rpc.pmap_rmt.h
/usr/include/rpc/pmap_rmt.h
/usr/include/rpc.raw.h
```

```
/usr/include/rpc/raw.h
/usr/include/rpc.rpc.h
/usr/include/rpc/rpc.h
/usr/include/rpc.rpcb_clnt.h
/usr/include/rpc/rpcb_clnt.h
/usr/include/rpc.rpcb_prot.h
/usr/include/rpc/rpcb_prot.h
/usr/include/rpc.rpcent.h
/usr/include/rpc/rpcent.h
/usr/include/rpc.rpc_com.h
/usr/include/rpc/rpc_com.h
/usr/include/rpc.rpc_mp.h
/usr/include/rpc/rpc_mp.h
/usr/include/rpc.rpc_msg.h
/usr/include/rpc/rpc_msg.h
/usr/include/rpc.svc.h
/usr/include/rpc/svc.h
/usr/include/rpc.svc_auth.h
/usr/include/rpc/svc_auth.h
/usr/include/rpc.svc_soc.h
/usr/include/rpc/svc_soc.h
/usr/include/rpc.types.h
/usr/include/rpc/types.h
/usr/include/rpc.xdr.h
/usr/include/rpc/xdr.h
/usr/include/sys.byteorder.h
/usr/include/sys/byteorder.h
/usr/include/sys.cmn_err.h
/usr/include/sys/cmn_err.h
/usr/include/sys.netconfig.h
/usr/include/sys/netconfig.h
/usr/include/sys.socket.h
/usr/include/sys/socket.h
/usr/include/sys.t_kuser.h
/usr/include/sys/t_kuser.h
/usr/include/sys.tiuser.h
/usr/include/sys/tiuser.h
/usr/include/sys.xti_inet.h
/usr/include/sys/xti_inet.h
```

3 Functions

The NSL-library contains the functions of the services TLI, XDR and RPC.

The following TLI-functions are supported:

```

int      t_accept ( int fd, int resfd, struct t_call *call );
char     *t_alloc ( int fd, inst struct_type, int fields );
int      t_bind ( int fd, struct t_bind *req, struct t_bind *ret );
int      t_close ( int fd );
int      t_connect ( int fd, struct t_call *sndcall,
                    struct t_call *rcvcall );
int      t_error ( char *errmsg );
int      t_free ( char *ptr, int struct_type );
int      t_getinfo ( int fd, struct t_info *info );
int      t_getprotaddr ( int fd, struct t_bind *boundaddr,
                        struct t_bind *peeraddr );
int      t_getstate ( int fd );
int      t_listen ( int fd, struct t_call *call );
int      t_look ( int fd );
int      t_open ( char *name, int oflag, struct t_info *info );
int      t_optmgmt ( int fd, struct t_optmgmt *req,
                    struct t_optmgmt *ret );
        unterstuetzte Optionen: TCP_NODELAY, SO_BROADCAST,
        SO_KEEPAALIVE;
int      t_rcv ( int fd, char *buf, unsigned nbytes, int *flags );
int      t_rcvconnect ( int fd, struct t_call *call );
int      t_rcvdis ( int fd, struct t_discon *discon );
int      t_rcvrel ( int fd );
int      t_rcvudata ( int fd, struct t_unitdata *unitdata,
                    int *flags );
int      t_rcvuderr ( int fd, struct t_uderr *uderr );
int      t_snd ( int fd, char *buf, unsigned nbytes, int flags );
int      t_snddis ( int fd, struct t_call *call );
int      t_sndrel ( int fd );
int      t_sndudata ( int fd, struct t_unitdata *unitdata );
int      t_sync ( int fd );
int      t_unbind ( int fd );

```

The XDR functions `xdr_string()`, `xdr_char()` and `xdr_u_char()` ENCODE to ASCII and DECODE to EBCDIC. The XDR functions `xdr_string_noc()`, `xdr_char_noc()` and `xdr_u_char_noc()` have the same functionality but without ASCII/EBCDIC conversion.

4 Restrictions

The entries of the functions are only available in upper case mode.

For the creation of the library the compiler option `ENUM-TYPE=LONG` (cc/c89 option `-XL ...`) was used. Programs, which use RPC or XDR functions, must be compiled with the same option. Otherwise the program may not run correctly (different alignment of the data).