SYSLIB

Release Notes

Applies to Product Release: 4.00.00.00-Alpha9
Publication Date: March 11, 2015

TEXAS INSTRUMENTS

Document License

Contents

Release Notes

SYSLIB 4.00.00.00-Alpha

1 INTRODUCTION

1.1 Overview

This document provides the release information for the SYSLIB software package. The SYSLIB package includes the following:-

- SYSLIB Release Notes
- SYSLIB User's Guide
- Source code of all SYSLIB components
- Pre-built libraries (Little Endian) of all SYSLIB components
- API reference guide
- Software Manifest

This is an engineering tested alpha release package. Release notes from previous releases are also available in the release notes archive directory

2 RELEASE OVERVIEW

2.1 Hardware Device Support

The device and platforms tested for this release include:

- TMDXEVM6638lxe

2.2 Components and Tools

The SYSLIB package is verified/tested using the MCSDK 3.01.02.05 package. Please refer to the MCSDK Release notes for a list of all the component information. The following is the list of additional packages which were used to test the release:

1. SNOW3G 1.0.0.2
2. CUIA 1.01.00.06 Custom

3. UIA 2_00_03_40_eng
4. SA3GPP Enabler 3.0.0.0

The SYSLIB supports only the RT kernel from the MCSDK release. Similarly please use the RT DEVKIT for the development of user space applications.

2.3 Licensing

Please refer to the software manifest

2.4 MCSDK Patches

Please ensure that the following environment variable is defined and saved in the UBOOT environment:-

```
setenv mem_reserve 1536M
```

This will ensure that the kernel reserved the higher order 1.5GB of memory for the DSP. Failure to do so will result in the kernel overwriting DSP memory.

The SYSLIB release modifies the default MCSDK released kernel DTS files. The kernel DTS files have been modified for the following features:-

- GIC Queues 8722 to 8735 were originally reserved for the Linux kernel. These queues are not used by the Linux kernel so these have been marked as unreserved and could not be used by the ARM applications
- Wiring of the GIC Queue and INTC_SET2 interrupt queues from using the UIO module.

Along with the kernel DTS file; the SYSLIB RMv2 files have also been modified for the following features:-

- GIC Queues 8722 onwards have been marked as usable
- INTC_SET2 queues have been allocated to ARM
- Wildcarding support

NOTE: Please integrate the SYSLIB released DTS files with your application and always update the kernel DTB files and SYSLIB RMv2 DTB files.

3 What's new

3.1 New Features

1. NETFP Master

The NETFP master is a common system wide application which has been created to initialize the NETCP and provide physical interface configurations at system start up time. The master provides the following features:-

1. Programmability of Reassembly feature; which allows the master to handle reassembly of fragments for the entire system
2. Programmability of enhanced QoS feature
3. On per physical interface base, we provide:
    a. Pre-classification configuration
    b. DSCP Marking, i.e. rules to create outer DSCP/pBit value
    c. L2 QoS configurations which includes:
        i. Support for multiple contiguous QoS channels and Flows
        ii. DSCP mapping to L2 QoS channel and flow
        iii. VLAN bPit mapping to L2 QoS channel and flow

The NETFP master also exposes messaging interface to support limited dynamicity. The NETFP master configuration parameters are exposed using a configuration file. Please refer to the `ti/apps/netfp_master/netfp.conf` for a sample file.

Invoke the NETFP master using the following command:-

```
./netfp_master_k2h.out -r Rm_System -c ./netfp.conf &
```

NOTE: It is required to start the NETFP Master before executing the NETFP Server. The NETFP master like the RM Server is a system wide application which needs to be executed before any RAT specific services are started.

2. Fast Path Wild Carding

In this release, Fast Path wild carding feature has been added. A wild carded Fast Path is created when IP address is set to all zeroes. From this release, LUT1-1 and LUT1-2 entries are managed through Resource Manager. This includes both normal

Fast Path and wild carded Fast Path. LUT entries for these two categories are reserved in DTS files.

In order to use this feature; while adding an Inbound Fast Paths an additional parameter ‘`spidMode`’ needs to be specified:-

SPID Mode	Description
`Netfp_SPIDMode_INVALID`	Use this for non-secure mode or if the Security policy identifier is not known.
`Netfp_SPIDMode_ANY_SECURE`	Use this for secure mode where traffic from any security policy identifier can be received by the fast path
`Netfp_SPIDMode_SPECIFIC`	Use this for secure mode if using a specific Security Policy identifier.

3. **NETFP Socket Notification Infrastructure**

   In the previous releases; on certain events if the NETFP Fast path became inactive it would cause the socket to become inactive too. There was no recovery from this point onwards and the socket and fast paths would need to be deleted and recreated again.

   The release upgrades the NETFP event management infrastructure which allows sockets and fast paths to move between the inactive and active states.

4. **FAPI Version upgrade**

   The FAPI tracing library delivered with alpha 9 is upgraded to FAPI v2.1.

5. **NETFP Ethernet Rule management**

   In this release, management (adding/deleting) of Ethernet Rules in PA LUT1-0 is added. Ethernet rules can be added/deleted through NETFP master’s messaging interface. The configuration of the Ethernet rule needs to specify the LUT index region from the dts file. One default region is added in the dts file as an example. An error will be returned if region name is not available in dts file. The regions can be extended if needed.

6. **NETFP Proxy routing flush command**

   To support dynamic route recalculation, Netfp proxy exports an IP route flush command. This command MUST be executed whenever the application modifies the IP routing table. The IPC message type to use is NETFP_PROXY_IPC_MSGTYPE_IP_ROUTE_FLUSH_REQ.

   Or execute “ip_route_flush” via the Netfp proxy command shell.

3.2 Bug Fixes

1. **RESMGR Source Identifier updated:**

   The previous SYSLIB release used IPC Source identifier 0 for Root and Source Identifier 1 for the SYSRM. These source identifiers conflicted with the MCSDK DSP-ARM download feature; so from this release onwards the following source identifiers are being used:

IPC Source Identifier	Description
16	Root
17	Resource Manager

Please ensure that the ***SYSRM server*** is now started with the proper configuration:-

```
./resmgr_server_k2h.out -n Rm_Server -g global-resource-list.dtb -s 17
-p policy_dsp_arm_syslib.dtb &
```

Similarly on the DSP please ensure that the ***SYSRM services*** are initialized with the proper configuration:-

```
    sysConfig.dspSystemCfg.armCoreId                  = 8;
    sysConfig.dspSystemCfg.sourceId                   = 17;
    sysConfig.dspSystemCfg.sharedMemAddress           = DDR3_SYSLIB_RESMGR_RSVD;
    sysConfig.dspSystemCfg.sizeSharedMemory           = DDR3_SYSLIB_RESMGR_RSVD_LEN;

    /* Initialize the system configuration. */
    handleSysCfg = Resmgr_init(&sysConfig, &errCode);
    ...
```

For the root; please ensure that the changes are reflected properly in the ***Root Master***

```
    rootMasterConfig.ipcSrcId                 = 16;        /* IPC Src Id: 16 */
    rootMasterConfig.osalFxn.malloc           = Root_osalMalloc;
    rootMasterConfig.osalFxn.free             = Root_osalFree;
    rootMasterConfig.osalFxn.beginMemAccess = Root_osalBeginMemoryAccess;
...
    /* Create the root master */
    rootMasterHandle = Root_masterCreate (&rootMasterConfig, &errCode);
    if (rootMasterHandle == NULL)
...
```

For each ***root slave***; please ensure that the following changes are done to reflect the new configuration:

```
    rootSlaveCfg.masterCoreId           = 8;
    rootSlaveCfg.ipcSrcId               = 16;
    rootSlaveCfg.osalFxn.malloc         = Root_osalMalloc;
...
    /* Create the root slave. */
    rootSlaveHandle = Root_slaveCreate (&rootSlaveCfg, &errCode);
    if (rootSlaveHandle == NULL)
...
```

3.3 API changes:

1. **SYSRM OSAL changes:**

   The MCSDK RMv2 introduces a new set of OSAL API which provides protection against concurrent access from multiple threads. In order to integrate these modifications into the SYSRM; the SYSRM OSAL API has been modified as follows:-

   - Removes the redundant: `enterCS and exitCS`
   - New API added: `createSem, deleteSem, postSem and pendSem`.

   Please refer to the OSAL implementations in the test projects for a sample implementation on ARM and DSP.

2. **NETFP Fast Path changes:**

   The table below summarizes the changes which were introduced:

Old API	New API	Description
`Netfp_addFastPath`	`Netfp_createInboundFastPath` `Netfp_createOutboundFastPath`	The old API was overloaded to handle both INBOUND & OUTBOUND fast path. NETFP now clearly differentiates between the two. The OUTBOUND Fast path is an asynchronous API because it performs next hop route resolution too. Once successfully completed

		the OUTBOUND fast path becomes active.
`Netfp_delFastPath`	`Netfp_deleteInboundFastPath` `Netfp_deleteOutboundFastPath`	The old API was overloaded to handle both INBOUND & OUTBOUND fast path. NETFP now clearly differentiates between the two.
`Netfp_findFastPath`	`Netfp_findInboundFastPath` `Netfp_findOutboundFastPath`	The old API was overloaded to handle both INBOUND & OUTBOUND fast path. NETFP now clearly differentiates between the two.
`Netfp_resolveRoute`	`N/A`	Route resolution is now not exported to the application. It is automatically initiated once an OUTBOUND fast path is created. The next hop MAC address is now stored on a per fast path basis.
`Netfp_isRouteResolved`	`Netfp_isOutboundFastPathActive`	As mentioned above; the creation of an OUTBOUND fast path initiates the next hop resolution process. The API can be used to determine if the resolution was completed or not.

3. **NETFP Socket Address:**

   The NETFP socket address structure has been modified to clearly differentiate between the `bind` and `connect` operations.

4. **NETFP Socket Notification change:**

   The NETFP socket notification function definition has been modified as follows:

```
void (*Netfp_NotifyFunction)
(
    Netfp_Reason reason,
    Netfp_SockHandle sockHandle
);
```

The function now notifies the socket with a reason code. Please refer to the NETFP header file to get an updated list of reasons.

5. **NETFP Get Payload change:**

   The NETFP Get Payload function has been modified as follows:

```
int32_t Netfp_getPayload
(
    Ti_Pkt*             ptrRxPkt,
    uint8_t             gtpuPayload,
    uint8_t**           ptrPayload,
    uint32_t*           payloadLen,
    Netfp_PeerSockAddr* ptrPeerAddress
)
```

   An additional parameter `gtpuPayload` has been added. The parameter serves as a hint which allows the application to specify if the function to skip the GTPU header or not. The GTPU Header is not skipped in all cases (For example: If the UDP Destination port is not 2152 or if the GTPU message type is not 0xFF)

   The return value has also been extended to return the status on how many headers have been skipped.

6. **NETFP Proxy Interface name change:**

   NETFP Proxy no longer supports specifying the interface name during offloads or creating interfaces during netfp proxy plugin initialization. Instead the interface to be created is computed when the policy is offloaded in case of secure policies. For non-secure policies, the interface is created when the outbound fast path is created.

3.4 Feature list in JIRA:

The release feature list accounted for in TI JIRA database:

Issue Type	Key	Summary	Requirement
Bug	SCLTE-1845	Syslib4 listlib definition conflict	
Bug	SCLTE-1885	Incorrect descriptor handling in NetfpMaster when re-assembly is disabled	
Bug	SCLTE-1682	Netfp Proxy crashes (dumps core) when route resolve is invoked after ip address is deleted and added.	
Story	SCLTE-1647	Support for K2K needed in syslib	SCLTE-1611
Story	SCLTE-1793	LUT1-0 Entries Management	SCLTE-1675
Story	SCLTE-1227	Cascading operation	SCLTE-366
Story	SCLTE-1221	Interface based cascaded traffic shaping	SCLTE-1114
Story	SCLTE-1843	Integrate FAPI Trace Library with FAPI 2.1	
Story	SCLTE-1800	NetFP support of port capturing	SCLTE-1720
Story	SCLTE-1220	Egress shaping using both L3 and L2 QoS	SCLTE-1045
Story	SCLTE-1208	NetFP server - NetFP clients notifications	SCLTE-1037
Story	SCLTE-1213	NetFP client shall inform an application in case of socket/channels becoming not operational	SCLTE-1042
Story	SCLTE-1212	NetFP client shall handle the notifications of routing and MTU changes without disturbing an application	SCLTE-1041
Story	SCLTE-1431	NetFP Networking: Ingress helper function	SCLTE-1415
Story	SCLTE-1799	NetFP support of port mirroring	SCLTE-1713
Story	SCLTE-1830	NetFP handling of socket object during dynamic updates	SCLTE-1829
Bug	SCLTE-1718	Route delete on Non-Default route with gateway IP as zero does not remove this non-default route from IPV4 route table	
Bug	SCLTE-1683	switchPortNum parameter needs to be fixed in Netfpproxy	
Bug	SCLTE-1374	Overlapping parameters in DTS causes errors in RemoteProc driver	
Story	SCLTE-1847	NetFP support of UDP Sockets bound to port 2152	SCLTE-1808
Bug	SCLTE-1674	Memory leak in syslib4 dat_uio.c	
Story	SCLTE-1230	IPSec re-keying	SCLTE-402
Bug	SCLTE-1681	Netfp_delInterfaceIP() function does not clear the isValid status	
Story	SCLTE-1432	NetFP Networking: NetCP pre-classification configuration	SCLTE-1425
Story	SCLTE-1429	NetFP Networking: L2 QoS static configuration	SCLTE-1422
Bug	SCLTE-1552	FAPI Tracing - potential crash when pktlibInstHandle and pktlibHeapHandle uninitialized	
Story	SCLTE-1511	Sync to MCSDK 3.1.2	
Story	SCLTE-1109	Policy Offload. Number of Physical Interfaces	SCLTE-920

Story	SCLTE-1216	Egress shaping of VLAN traffic	SCLTE-1139
Story	SCLTE-1217	Interface based egress traffic shaping	SCLTE-1047
Story	SCLTE-1427	NetFP Networking: Fast Path packet routing through L3 QoS channels	SCLTE-1424
Story	SCLTE-1430	NetFP Networking: Fast Path packet marking	SCLTE-1423
Story	SCLTE-1324	Support in NetFP_Proxy to offload multiple policies using the same SA	SCLTE-130
Story	SCLTE-1369	Fast Path Wildcarding	SCLTE-348
Story	SCLTE-1526	Fast Path Policy wildcarding	SCLTE-1510

3.5 Known Issues:

Issue Type	Key	Priority	Summary
Bug	SCLTE-1795	Major	Default route with Gateway IP as zero is not supported correctly in NetfpProxy
Bug	SCLTE-1612	Minor	while(1) loop in msgcom code needs to be removed.
Bug	SCLTE-1370	Major	Syslib 4 unitests do not pass out of the box.
Bug	SCLTE-1892	Major	Kernel Crash [UIO Module]
Bug	SCLTE-1898	Major	LTE Demos Not Supported

4 RELEASE BUILDING

Please setup the following environment variables:-

```
export
ARMTOOLS_INSTALL_PATH=/home/a0868491/tools/gcc-linaro-arm-linux-gnueabihf-4.7-
2013.03-20130313_linux

export
ARAGO_INSTALL_PATH=/home/a0868491/ti/mcsdk_linux_3_01_02_05_devkit_rt/sysroots
/cortexa15hf-vfp-neon-3.8-oe-linux-gnueabi

export CGT_INSTALL_PATH=/home/a0868491/ti/cgt_7.4.8

export XDC_INSTALL_PATH=~/ti/xdctools_3_30_04_52

export PDK_INSTALL_PATH=~/ti/pdk_keystone2_3_01_02_05/packages

export SNOW3G_INSTALL_PATH=~/ti/snow3g_1_0_0_2/packages

export UIA_INSTALL_PATH=~/ti/uia_2_00_03_40_eng/packages

export INSTALL_JAMMER_INSTALL_PATH=~/tools/installjammer-1.2.15
export BIOS_INSTALL_PATH=~/ti/bios_6_40_04_47/packages
export IPC_INSTALL_PATH=~/ti/ipc_3_30_01_12/packages
export CUIA_INSTALL_PATH=~/tools/cuia_1_01_00_06Custom
export SYSLIB_DEVICE=k2h
export SYSLIB_INSTALL_PATH=~/ti/syslib_4_00_00_00_alpha9/packages
```

The environment variables are illustrative and should be modified by the customer as per their install paths.

Once configured please setup the build environment by executing the following script:-`SYSLIB_INSTALL_PATH/scripts/setupenv.sh`. This will setup the build environment and will also sanity check to make sure that all the required environment variables are configured.

Please execute the release script (`SYSLIB_INSTALL_PATH/scripts/release.sh`) as follows:-

```
    source ./release.sh 1 0 1
```

The script takes 3 arguments:-

1. Argument1: Build the SYSLIB ARM libraries.
2. Argument2: Build the SYSLIB DSP libraries. [Should always be set to 0]
3. Argument3: Build the SYSLIB LTE Demo

To rebuild the DSP libraries; please execute the following path from the SYSLIB_INSTALL_PATH

```
xdc clean –PR .
xdc –PR .
```

For information on how to build the DSP and ARM unit tests and for execution instructions please refer to the SYSLIB Unit Test documentation.