SysLink Install Guide

Introduction

This SysLink Install Guide provides basic information on how to get started with the SysLink product. It contains the necessary steps on how to install and build SysLink, its dependencies (where relevant), and the SysLink examples. Information on how to install (on the devices target file-system) and run the SysLink examples is also included.

See the SysLink Release Notes for details about devices and component versions supported in each SysLink release. This guide is appropriate for the SysLink 2.20 release, and compatible. For older releases, see the SysLink Install Guide - 2.00 and 2.10.

The SysLink product consists of three main components for any supported device:

1. Master core driver (executable) that is running a high-level OS (HLOS GPP driver)
2. Master core user level libraries (HLOS GPP libraries)
3. Slave core(s) user level libraries that is running a Real-time OS (RTOS slave libraries).

SysLink applications are typically written to interface with the GPP libraries interface on the master core and the slave libraries on the slave core(s). See the SysLink User's Guide for more information.

Install

Basic installation

SysLink is released as a tar.gz file. To install, simply extract the file.

```
buildhost$ tar -xzvf syslink_<version>.tar.gz -C /home/user
```

This will extract the SysLink product in a directory with its product name and version information (e.g. /home/user/syslink_2_xx_xx_xx)

NOTE

- This document assumes the install path to be the user's home directory on a Linux host machine. This path (**/home/user**) will be used throughout the document. If SysLink was installed at a different location, make appropriate changes to the commands.
- QNX users: if you use a Windows build host machine, simply use your favorite program (e.g. Winzip) to uncompress the tar.gz file into a directory of your choice (e.g. C:\ti) instead of **/home/user**
- Some customers find value in archiving the released sources in a configuration management system. This can help in identifying any changes made to the original sources - often useful when updating to newer releases.

Dependencies

The dependencies applicable to each device can be found in the SysLink Release Notes provided in the product.

Build

The SysLink product provides a top level GNU makefile to allow users to easily build the various component modules and sample applications for the supported devices. GNU make version 3.81 or greater is required. The XDC tools (provided with most SDKs and CCS distributions) includes a pre-compiled version of GNU make 3.81 in $(XDC_INSTALL_DIR)/gmake.

Setup

The SysLink install directory (**/home/user/syslink_2_xx_xx_xx**) contains a file (products.mak) that specifies the necessary paths and options to build the SysLink for a specific device.

- Edit **products.mak** and Save.
  - Set the **DEVICE** variable to a supported device.
  - Configure the various build #Options to build SysLink - (GPPOS, EXEC_DIR, etc.).
  - Set the required device paths to the device's dependent components and tools - (IPC_INSTALL_DIR, BIOS_INSTALL_DIR, etc.).

You can issue the *.show-products* goal to print your settings.

```
buildhost$ make .show-products
```

Options

This section contains a descriptions of the various SysLink options exposed to the user through the top level GNU makefile in the products.mak file.

- DEVICE - Device to be built for.
- GPPOS - Set the devices master core(GPP) OS type.
- SysLink GPP driver build options
  - SYSLINK_TRACE_ENABLE - SysLink provides a mechanism to enable or disable different levels of trace prints on both the kernel and user side.
  - SYSLINK_BUILD_DEBUG - SysLink also has a mechanism to display assertion prints when an unexpected condition is seen.
  - SYSLINK_BUILD_OPTIMIZE - SysLink provides a mechanism to substantially optimize the SysLink code through a compile option by removing checks for error conditions, invalid parameter checks and intermediate failures.

    See the SysLink User's Guide for a complete description on these build settings.
- SysLink GPP driver build option for selection of the Notify driver and MessageQ transport (see here for more detail.)
  - SYSLINK_NOTIFYDRIVER - SysLink option for selection of the Notify driver.
  - SYSLINK_TRANSPORT - SysLink option for selection of various MessageQ transports.
- SDK - Set SDK type when building for a device that is supported by TI's SDK development kits or NONE.
- EXEC_DIR - Set the base directory for the SysLink kernel module and examples to be installed to for running on the devices target file-system.
- LOADER - The loader type needed to load the slave (DSP, M3, etc.) executable (default in bold).
  - TI816X (**ELF**)
  - TI814X (**ELF**)
  - OMAP3530 (**ELF** or COFF) - note, COFF support is deprecated in SL 2.20
  - OMAP-L138 (**ELF** or COFF) - note, COFF support is deprecated in SL 2.20
- TI81XXDSP_DMTIMER_FREQ - This must match the frequency of the Dmtimer (applies to TI81XX only). Common values are 32768 or 20000000.

Build SysLink Driver and Libraries

After updating **products.mak** for your environment, you can build the SysLink driver and libraries. Issue the following command in the SysLink install directory:

```
buildhost$ make syslink
```

NOTE
If desired, you can optionally build parts of SysLink individually (GPP driver, GPP user libraries, RTOS libraries) with these build goals:

```
buildhost$ make syslink-driver
buildhost$ make syslink-hlos
buildhost$ make syslink-rtos
```

Depending on your build settings, some of these build goals may not build anything. For example when GPPOS is set to Bios, building the syslink-hlos goal will build nothing.

The user can override some (or all!) of the variables defined in the **products.mak** file using the command line, as follows:

```
buildhost$ make DEVICE=TI816X \
           GPPOS=Linux \
           LOADER=ELF \
           SDK=EZSDK \
           SYSLINK_INSTALL_DIR=/home/user/syslink_x_xx_xx_xx \
           IPC_INSTALL_DIR=/home/user/ipc_x_xx_xx_xx \
           BIOS_INSTALL_DIR=/home/user/bios_x_xx_xx_xx \
           XDC_INSTALL_DIR=/home/user/xdctools_x_xx_xx_xx \
           LINUXKERNEL=/home/user/TI816X-LINUX-PSP-xx.xx.xx.xx/src/kernel/linux-xx.xx.xx.xx \
           CGT_ARM_INSTALL_DIR=/home/user/arm/arm-xxxxqx-xxx \
           CGT_ARM_PREFIX=/home/user/arm/arm-xxxxqx-xxx/bin/arm-none-linux-gnueabi- \
           CGT_C674_ELF_INSTALL_DIR=/home/user/c6x/x.x.x/ \
           syslink
```

Build SysLink Examples

Introduced in SysLink 2.20, SysLink includes an **examples/archives** directory containing platform-specific examples. You can extract and build the SysLink examples appropriate for your environment with the following command in the SysLink install directory:

```
buildhost$ make examples
```

NOTE
The variable assignments in **products.mak** are used to determine which platform-specific examples will be extracted and built.

The user can override some (or all!) of the variables defined in the **products.mak** file using the command line, as follows:

```
buildhost$ make DEVICE=TI816X \
           GPPOS=Linux \
           LOADER=ELF \
           SDK=EZSDK \
```

```
            SYSLINK_INSTALL_DIR=/home/user/syslink_x_xx_xx_xx \
            IPC_INSTALL_DIR=/home/user/ipc_x_xx_xx_xx \
            BIOS_INSTALL_DIR=/home/user/bios_x_xx_xx_xx \
            XDC_INSTALL_DIR=/home/user/xdctools_x_xx_xx_xx \
            LINUXKERNEL=/home/user/TI816X-LINUX-PSP-xx.xx.xx.xx/src/kernel/linux-xx.xx.xx.xx \
            CGT_ARM_INSTALL_DIR=/home/user/arm/arm-xxxxqx-xxx \
            CGT_ARM_PREFIX=/home/user/arm/arm-xxxxqx-xxx/bin/arm-none-linux-gnueabi- \
            CGT_C674_ELF_INSTALL_DIR=/home/user/c6x/x.x.x/ \
            examples
```

NOTE

The initial SysLink 2.20 release doesn't contain any examples for an all-BIOS environment.

Most IPC features (e.g. MessageQ) are provided by the IPC product, and users should refer to the IPC product for those examples. The features SysLink provides above the IPC product include:

- ProcMgr - we will provide an example to load the slave in a future release
- RingIO - very few, if any, users of RingIO on the all-BIOS environment so no example is currently planned
- FrameQ - deprecated as of SL 2.20, so no example is currently planned.

Run

The SysLink makefiles provide a simple way to install (copy) the necessary SysLink executables (e.g. HLOS driver), examples and helper scripts onto the device's target file-system to simplify the execution of the applications. The details can vary across OS's, so this description has been separated into OS-specific sections.

Linux

Configuring Kernel Parameters

SysLink requires a few specific arguments to be passed to the Linux kernel during boot up. To run the examples, 3MB of memory is needed by SysLink for communication between GPP and slave(s), and for slave(s) external memory usage to place its code/data sections. This must be reserved by specifying 3MB less of available memory for Linux kernel usage.

- For example, with available memory of 256M, memory required for shared regions/other utils is 5M and SysLink is 3M leaving only 248M for the Linux kernel.

```
bootargs console=ttyS2,115200n8 root=/dev/nfs nfsroot=HOST:nfs_root,nolock rw mem=248M ip=dhcp
```

This is just an example, bootargs may vary depending on available setup

Depending on the memory map used in the final system configuration, the memory to be reserved for Linux kernel usage may differ.

Installing Examples

To assemble the SysLink executables and examples into a directory structure suitable for running on the device's file-system, issue the following command in the SysLink install directory:

```
buildhost$ make install
```

NOTE
The examples will be installed in the EXEC_DIR directory set in the **products.mak** file. As with other variables, you an override this on the command line:

```
buildhost$ make install EXEC_DIR=/home/user/ti816x/filesystem
```

SysLink Kernel Driver

The SysLink kernel driver must be inserted into the kernel for SysLink user applications to run.

Loading module

The kernel module is located in the /lib/modules/<kernel_version>/kernel/drivers/dsp directory on the device's target file-system if the EXEC_DIR path in products.mak was set to the root of the device's target file-system.

If the SysLink GPP driver is built with trace enabled, the driver needs to enable both TRACE and TRACEFAILURE to configure SysLink to print out kernel messages in case any failures occur during SysLink user application runs. To load the module issue the following command on the target file-system:

```
target# insmod /lib/modules/<kernel_version>/kernel/drivers/dsp/syslink.ko TRACE=1 TRACEFAILURE=1
```

Unloading module

The kernel module can be unloaded by issuing the following command on the target file-system:

```
target# rmmod syslink
```

Running Examples

Each example contains a **readme.txt** and **run.sh** script to demonstrate how to run it.

The root of the samples directory on the target file-system (/ti/syslink-examples/<device>) contains a script to execute all the examples provided.

QNX

Installing Examples

To assemble the SysLink executables and examples into a directory structure suitable for running on the device's file-system, issue the following command in the SysLink install directory:

```
buildhost$ make install
```

NOTE
The examples will be installed in the EXEC_DIR directory set in the **products.mak** file. As with other variables, you an override this on the command line:

```
buildhost$ make install EXEC_DIR=/home/user/ti816x/filesystem
```

Then use the QNX IDE to copy the files to the target file system:

- $EXEC_DIR/ti/syslink/driver -> /sd/SysLink
- $EXEC_DIR/ti/syslink/lib -> /sd/SysLink

- $EXEC_DIR/ti/syslink/samples -> /sd/SysLink

SysLink Driver

The SysLink driver must be first run before any SysLink user applications can run.

On the target, add SysLink to the library search path. (Note this is only needed to run executables dynamically linked against the .so library):

```
target# export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/sd/Syslink/lib
```

Then launch the SysLink driver.

```
target# /sd/SysLink/driver/debug/syslink_drv
```

If the SysLink driver is built with trace enabled, the driver needs to enable both TRACE and TRACEFAILURE to configure SysLink to print out messages in case any failures occur during SysLink user application runs. To load the driver with trace output during failures, issue the following command on the target file-system:

```
target# export TRACE=1
target# export TRACEFAILURE=1
target# /sd/SysLink/driver/debug/syslink_drv
```

Alternatively, as a convenience, you can also run the script /sd/SysLink/driver/debug/run_syslink_drv.sh to achieve the same result.

The driver can be unloaded by issuing the following command on the target file-system:

```
target# slay syslink_drv
```

Running Examples

Each example contains a **readme.txt** and **run.sh** script to demonstrate how to run it.

The root of the samples directory on the target file-system (/ti/syslink-examples/<device>) contains a script to execute all the examples provided.

Slaveloader

For HLOS's, the SysLink product provides a 'slaveloader' utility to load, start and stop the slave processor. In addition to demonstrating how to manage slave devices, many of the examples on HLOS's leverage the slaveloader utility to load the slaves prior to running the example.

The slaveloader utility takes up to four parameters:

- A loader command
- A remote processor name
- An executable name - **Required** when the loader command is **startup, load,** or **all**
- A map file - **Required** if the remote core's MMU is enabled (See the SysLink MMU Support article for more information)

```
slaveloader <startup|shutdown|all|powerup|load|start|stop|unload|powerdown|list> <Core name>
    [File path] [map-file]
```

- On providing the **startup** command, the slaveloader will powerup-load-start the slave core.
- On providing the **shutdown** command, the slaveloader will stop-unload-powerdown the slave core.

You can review the various example-specific **run.sh** scripts for slaveloader usage.

See Also

- For older releases, see the SysLink Install Guide - 2.00 and 2.10

Article Sources and Contributors

SysLink Install Guide *Source*: http://processors.wiki.ti.com/index.php?oldid=164121 *Contributors*: A0792201, Arnier, ChrisRing, Joerngr, PagePusher, RobTivy

License

matter take effect in the relevant jurisdiction in which the License terms are sought to be enforced according to the corresponding provisions of the implementation of those treaty provisions in the applicable national law. If the standard suite of rights granted under applicable copyright law includes additional rights not granted under this License, such additional rights are deemed to be included in the License; this License is not intended to restrict the license of any rights under applicable law.