SysLink FAQs

Contents

Generic questions

Where do I get SysLink releases?

- SysLink releases can be downloaded from here (http://software-dl.ti.com/dsps/dsps_public_sw/sdo_sb/targetcontent/syslink/index.html)

What do I get with SysLink, and how do I get started?

Each SysLink release is available as a full source release including:

- Complete source code for the SysLink product, with associated build system
- Sample applications demonstrating the usage of all modules.
- Release documents: Release Notes, User Guide, Install Guides

To get started with SysLink, you can refer to the release documents mentioned in the release notes in the base of the product. Refer to the SysLink User Guide for high level information. The SysLink API Reference Guide (http://software-dl.ti.com/dsps/dsps_public_sw/sdo_sb/targetcontent/syslink/latest/docs/html/index.html) provides details on the APIs, and the samples provided with SysLink demonstrate usage of the APIs.

The SysLink Install Guide provides installation and build instructions, as well as details on running the sample applications.

The Migration Guide gives migration information between DSPLink and SysLink.

What is the relation between SysLink and IPC software packages

The SysLink and IPC product are complementary.

- IPC includes the implementation independent headers (e.g. ti/ipc/MessageQ.h), as well as an implementation of that spec for BIOS (e.g ti/sdo/ipc/MessageQ.c)
- SysLink provides Linux-side implementation of the IPC interfaces, as well as additional higher level data transports such as RingIO and FrameQ for both BIOS and HLOS environments. (Note: FrameQ has been deprecated as of 2.20. Customers are encouraged to implement equivalent functionality using lower-level SysLink primitives).

If you are using Linux on the ARM and BIOS on a slave processor, then both SysLink and IPC packages needs to be used, since SysLink contains the required Linux-side components and IPC contains the BIOS side.

How does DSP/Link relate to SysLink/IPC packages?

Please refer to the following link: http://processors.wiki.ti.com/index.php/SysLink_Overview#How_is_SysLink_different_from_DSPLink

How to migrate from DSPLink to SysLink?

Please refer the SysLink Migration Guide for more information.

Build questions

Can I build both HLOS and RTOS sides of SysLink only on a Linux machine?

Yes! The SysLink make system supports build of both GPP and DSP-side on a Linux machine. In fact, it is advised to use this method when you are using Linux as the HLOS because:

- A single SysLink source base can be maintained for HLOS and RTOS-sides, minimizing any mismatch issues.
- Scripts can be written over basic SysLink make system for doing a single-shot build of both HLOS and RTOS-sides

- An NFS-mounted target file system can be easily used to transfer both generated HLOS and RTOS-side binaries into the target file system.

Can I build SysLink RTOS side on Windows Machine?

Yes! The SysLink RTOS side code, including sample applications, can be built on a windows machine. Details are in the SysLink Install Guide.

How do I build SysLink Samples(HLOS/RTOS) for a particular Platform?

See the SysLink Install Guide.

Build warnings/errors! Where do I look?

Check out the Troubleshooting SysLink Build Issues article. This lists most of the common build failures and their possible causes, and suggests solutions.

How do I reduce the memory used by IpcMemMgr in QNX?

In QNX, SysLink uses an internal module called IpcMemMgr to manage inter-process shared memory (ie. memory shared between local QNX processes that use SysLink). By default, this module conservatively allocates a 16 MB heap to store state information for other SysLink modules that are in use. In many systems, most of this memory is unused, and the heap can be resized once you have a working system. To do so, follow these steps to experimentally determine the size that is required:

- Add a printf call in IpcMemMgr_alloc() in <SYSLINK_INSTALL_DIR>/packages/ti/syslink/utils/hlos/knl/IpcMemMgr.c:

<syntaxhighlight lang="cpp">

```
      adjSize = (UInt32) size;
```

```
      /* Make size requested a multiple of obj->minAlign */
      if ((offset = (adjSize & (IpcMemMgr_module->global->minAlign - 1)))
          != 0) {
          adjSize = adjSize + (IpcMemMgr_module->global->minAlign - offset);
      }
      printf("IpcMemMgr_alloc is allocating size of %d\n", adjSize);    /* !!! add this line !!! */
```

</syntaxhighlight>

- Save the file. Rebuild the SysLink driver.
- Replace the SysLink driver (syslink_drv) on the target filesystem with the new one.
- Run the user application(s) that use SysLink. If you have more than one that you expect to run concurrently, make sure you do so to reproduce the use case with worst-case heap usage. By running the user application(s), you can see every allocation through the IpcMemMgr on the console. Adding the sizes up should give you an estimate of how much memory your application(s) needs from IpcMemMgr. The total size may be different for different applications, depending on which modules are used and how they are configured.
- Change the amount of memory allocated by IpcMemMgr to a size that is 2x the estimate obtained to account for potential fragmentation (plus any additional space for in case your applications change over time, at your discretion). This is done by modifying this line, located near the beginning of the IpcMemMgr.c file:

```
#define IpcMemMgr_SHAREDHEAP_SIZE   (16*1024*1024)
```

- Remove the printf statement from the SysLink driver and rebuild the latter.
- Finally run the user application(s) against the modified SysLink driver to use the new heap size. Verify that it still works.

Runtime questions

DSP hangs or Ipc_control(STARTCALLBACK) fails on TI81XX device

This could be a DM Timer misconfiguration.

Depending on the configuration, you may be able see a error message in the trace log. Attach to the DSP with CCS, point a memory window at __CIOBUF_ and change display to ascii characters.

To change dmtimer configuration, add the following to your DSP configuration file (cfg): <syntaxhighlight lang='javascript'> var Timer = xdc.useModule('ti.sysbios.timers.dmtimer.Timer'); Timer.intFreq.hi = 0; Timer.intFreq.lo = 20000000; </syntaxhighlight>

Ipc_control(STARTCALLBACK) timeouts on TI81XX device when only the M3 Core 1 is used

In general, Ipc_control(STARTCALLBACK) timeouts indicate the fact that the slave core did not complete the expected handshake. It could be due to a variety of factors that ultimately leads to the handshake code either not getting executed or failing to run correctly. E.g. the slave not coming out of reset, getting 'stuck' in startup code that gets run before Ipc_start() and Ipc_attach(), or failing to call Ipc_start()/Ipc_attach() correctly, etc.

On TI81xx devices, one common user error we have seen is if you are only loading and using the M3 (media controller) core 1 and not core 0, Ipc_control(Ipc_CONTROLCMD_STARTCALLBACK) for that core may timeout due to the slave being held in an infinite loop in Ipc_start() waiting to enter a gate via the function GateDualCore_enter(), thus failing to complete the handshake. In this case, you need to add the following lines to the configuration (.cfg) file of M3 core 1 to ensure that the gates shared by the M3's are properly initialized:

<syntaxhighlight lang='javascript'> var GateDualCore = xdc.useModule('ti.sysbios.family.arm.ducati.GateDualCore'); GateDualCore.initGates = true; </syntaxhighlight>

This initialization normally happens on M3 core 0 by default, but given core 0 is not used, it needs to be done on core 1.

Error Codes

Some common error codes and reasons you may see them:

- 0x83046005 - OSALKFILE_E_FILE_OPEN - Often seen if the slave executable to be loaded can't be found. Be sure the file exists and is readable. Also, try using a full path.

{{

1. switchcategory:MultiCore=

- For technical support on MultiCore devices, please post your questions in the C6000 MultiCore Forum
- For questions related to the BIOS MultiCore SDK (MCSDK), please use the BIOS Forum

Please post only comments related to the article **SysLink FAQs** here.

Keystone=

- For technical support on MultiCore devices, please post your questions in the C6000 MultiCore Forum
- For questions related to the BIOS MultiCore SDK (MCSDK), please use the BIOS Forum

Please post only comments related to the article **SysLink FAQs** here.

C2000=*For technical support on the C2000 please post your questions on The C2000 Forum. Please post only comments about the article **SysLink FAQs** here.*

DaVinci=*For technical support on DaVincoplease post your questions on The DaVinci Forum. Please post only comments about the article **SysLink FAQs** here.*

MSP430=*For technical support on MSP430 please post your questions on The MSP430 Forum. Please post only comments about the article **SysLink FAQs** here.*

OMAP35x=*For technical support on OMAP please post your questions on The OMAP Forum. Please post only comments about the article **SysLink FAQs** here.*

OMAPL1=*For technical support on OMAP please post your questions on The OMAP Forum. Please post only comments about the article **SysLink FAQs** here.*

MAVRK=*For technical support on MAVRK please post your questions on The MAVRK Toolbox Forum. Please post only comments about the article **SysLink FAQs** here.*

*For technical su please post you questions at http://e2e.ti.com Please post on comments abo article **SysLink** here.*

}}

Links

Amplifiers & Linear
Audio
Broadband RF/IF & Digital Radio
Clocks & Timers
Data Converters
DLP & MEMS
High-Reliability
Interface
Logic
Power Management
Processors

- ARM Processors
- Digital Signal Processors (DSP)
- Microcontrollers (MCU)
- OMAP Applications Processors

Switches & Multiplexers
Temperature Sensors & Control ICs
Wireless Connectivity