IPC LAB 2

ex02_messageq
Overview

• This is a MessageQ example using the client/server pattern.

• Topics covered in this example
  • Using SYS/BIOS heap for message pool
  • Anonymous message queue
  • Return address
This is a two processor example. It only builds for the HOST and DSP processors. 
  • You can build for either DSP1 or DSP2.

The DSP processor is the server. 
  • The server creates a named message queue. 
  • The server does not open any queues. 
  • The server returns all messages to the sender.

The HOST processor is the client application. 
  • The client creates an anonymous message queue 
  • The client creates and manages the message pool. 
  • The client sets the return address in the message header.
Data Flow

HOST

client

NULL: MessageQ

msg

msg

DSP1

Reserved Header

Heap

server

ServerQ: MessageQ

msg

msg

1

2

3

4

5
Step 1 — Work Area

• Create a work folder for this lab
  \texttt{C:\Ti\_Demo}

• Extract the example into the work folder
  <\texttt{ipc\_3\_30\_pp\_bb}>\texttt{\examples\DRA7xx\_bios\_elf\ex02\_messageq.zip}
Step 2 — Build Environment

• Set the product install paths as defined by your physical environment.
  
  • Edit `ex02_messageq/products.mak`
    
    ```
    DEPOT       = C:/Products
    IPC_INSTALL_DIR  = $(DEPOT)/ipc_m_mm_pp_bb
    BIOS_INSTALL_DIR = $(DEPOT)/bios_m_mm_pp_bb
    XDC_INSTALL_DIR  = $(DEPOT)/xdc tools_m_mm_pp_bb
    ```

• Set the tool paths (only need the ones you actually plan to use).
  
  • Edit `ex02_messageq/products.mak`
    
    ```
    gnu.targets.arm.A15F  = $(DEPOT)/gcc_arm_none_eabi_m_m_p
    ti.targets.elf.C66    = $(DEPOT)/ti_c6x_m_m_p
    ```

• Each example has its own `products.mak` file; you may also create a `products.mak` file in the parent directory which will be used by all examples.
Step 3 — Build Executables

• Open a Windows Command Prompt
  
  Start > Run
  cmd

• TIP: Use the following command to create an alias for the make command
  
  doskey make="C:\Products\xdctools_3_30_04_52\gmake.exe" $

• Build the example
  
  cd ex02_messageq
  make

• The executables will be in their respective "bin" folders
  
  ex02_messageq\dsp1\bin\debug\server_dsp1.exe
  ex02_messageq\host\bin\debug\app_host.xa15fg
Step 4 — Load Processors (1/2)

- Load the HOST processor
  - Debug view > CortexA15_0 > Select
  - Run > Load > Load Program
  - Click Browse, select the HOST executable
    C:\TI_Demo\ex02_messageq\host\bin\debug\app_host.xa15fg

- Run HOST processor to main (if needed)
  - CortexA15_0 > Select
  - Run > Go Main

- Set a breakpoint in App_delete
  - Disassembly view > textbox > App_delete
  - Double-click in margin (Ctrl+Shift+B) to set breakpoint
Step 4 — Load Processors (2/2)

- To load the DSP processor, you must have the host processor running (when Auto Run to Main is enabled).
  - Run the HOST processor. It will be spinning in the `Ipc_attach` loop.

- Load the DSP1 processor
  - Debug view > C66xx_DSP1 > Select
  - Run > Load > Load Program
  - Click Browse, select the DSP1 executable
    
    C:\TI_Demo\ex02_messageq\dsp1\bin\debug\server_dsp1.xe66

- Run DSP processor to main (if needed)
  - C66xx_DSP1 > Select
  - Run > Go Main
Step 5 — Run to Completion

• Set a breakpoint in
• Run the DSP. The test completes very quickly.
• Halt the DSP and then the HOST.
  • It is best to always halt the HOST last because this will also top the timers from running.
**ROV — LoggerBuf Module**

- Use the RTOS Object Viewer (ROV) to inspect the LoggerBuf module.
  - Debug view > C66xx_DSP1 > Select
  - RTOS Object View (ROV) > LoggerBuf > Select
  - Records (tab) > Select
  - AppLog > Select

- You will see a list of log events.
Congratulations!
End of Lab 2