A UDP Server for the XPS_iBOB_base system

Andrew Siemion, Mark Wagner

March 29, 2009

1 Introduction

This update to the XPS iBOB base system replaces the TCP server instantiated when including a LWIP yellow block with a UDP server to improve performance, robustness, and the PPC memory footprint of a given design. In addition to the UDP stack, this version of IBOB base system software includes a preprocesser conditional directive '`MAKESMALL`' to vigorously remove unnecessary features from TinySH (help text, auto-completion, etc..) for situtations where codespace is at a premium. To activate it, add the `-DMAKESMALL` option to the `ProgCCFlags:` line in your design's `XPS_iBOB_base/system.xmp` file or prepend '`#define MAKESMALL`' to your design's `XPS_iBOB_base/Software/main.c` file and recompile the design's C source (re-run the Software Generation step in `bee_xps`).

2 Installation

The UDP version of the base system is not in the current version of the casper libraries, as it is not backwards compatible. In order to install it, you must download the UDP version of the XPS_iBOB_base.zip from the casper wiki in the 'Downloads' section:

`HTTP://Casper.berkeley.edu/wiki/index.php/Downloads`

And then replace the existing version with the new one. By examining the batch script you are using to start Matlab/Simulink, you should be able to see the path to the casper library directory you are using (e.g. mlib_devel_7_1). The batch script is probably setting the global variable `MLIB_ROOT` to this directory. Inside that directory is a sub directory named `xps_lib`. Open up `xps_lib` and replace the current `XPS_iBOB_base.zip` with the downloaded one. If you want to support both versions of the base system, you'll need to have two separate check outs of the casper library and two separate Matlab batch scripts to access them.

3 UDP Server Instructions:

The IBOB UDP Server REPLACES the TCP telnet server usually instantiated in IBOB builds that include the CASPER LWIP yellow block. The IBOB UDP Server operates on UDP port 7 and processes and responds to TinySH commands sent within UDP packets.

3.1 Text mode

In the simplest operational mode, users send a UDP 'text command request packet' containing a valid TinySH command terminated with a line feed, or a sequence of line feed-terminated commands, (A line feed is ASCII code 0x0A or a '\n' in C-like programming languages) to an IBOB UDP Server. The IBOB UDP Server will respond with UDP 'text command output packet(s)' containing up to 1024 bytes of text output from TinySH, sent to the sending port used for the original text command request packet. The maximum UDP payload size of text command output packets is 1024 bytes. Multiple packets will be sent, if necessary. Below is example usage using the `nc` (or `netcat`) utility, where the IBOB is configured to use IP `192.168.0.10`:

Execute the 'listdev' command:

```
echo "listdev \n" | nc 192.168.0.10 7 -w 1
```

Execute two commands at once:

```
echo "regread reg1 \n regread reg2 \n" | nc 192.168.0.10 7 -w 1
```

Note that the `-w` flag is to prevent `nc` from hanging. In some versions of `nc` the timeout flag is different so it may behoove you to read the `man` page.

3.2 Robust Mode (for programattic applications)

Users can also send a UDP 'robust command request packet' containing a valid 8 byte 'robust command header' (see below) followed by a line feed-terminated TinySH command, or sequence of line feed-terminated commands, to an IBOB UDP Server. The maximum length of a robust command request packet is 1000 bytes.

The IBOB UDP Server will respond with robust command output packet(s), each containing an identically formatted 8 byte header followed by up-to 1024 bytes of text output from TinySH, sent to the sending port used for the original command. The maximum UDP payload size of robust command output packets is 1032 bytes.

The 'Session Identifier' field in the robust command request packet can optionally be set to a unique 3 byte command sequence identifier, which will be echoed back to the sender in the subsequent robust command output packets. The remaining fields in the header are ignored for robust command request packets.

Robust command output packets that are part of a sequence of packets, but not the last packet, will have a Type value of 1. A single command output packet, or the last command output packet in a sequence of packets, will have a Type value of 2. Robust command output packet's 'Sequence Number' field will increment from 0, 1, 2 ... 65535. If the total length of the text output from TinySH is a multiple of 1024 bytes, an additional 8 byte header-only packet will be sent indicating command termination.

	Robust Command Header Structure							
Byte	0	1	2	3	4	5	6	7
Field	255	Session ID			Session #		Type	Unused

All robust command header fields are unsigned integers. For additional questions, email `mwagner@eecs.berkeley.edu` or `siemion@berkeley.edu`.