Evaluation of TIPC

Mitsuhiro Kimura
TOSHIBA CORPORATION

CELF Japan Technical Jamboree #37, May 20, 2011
Background

• Integrating one more SoC in the embedded system for adding more functions.
• We will discuss the interconnection of two SoCs:
  – It needs to choose a bus from available interface buses in SoCs.
  – Ethernet is more commonly available.
  • Required a quick and lightweight protocol stack on Ethernet.
TIPC : Transparent Inter-Process Communication

• The protocol specifically designed for cluster communication – quick and lightweight
• Supposing little packet loss and infrequent retransmission
  – Socket I/F
    • sockets corresponding TCP, UDP.
  – Built in Linux Kernel.

<table>
<thead>
<tr>
<th>TIPC:</th>
<th>1.4.3</th>
<th>1.5.12</th>
<th>1.6, 1.7</th>
<th>2.0.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kernel:</td>
<td>2.4, 2.6</td>
<td>2.6.9 ~ 14</td>
<td>2.6.16 ~ 34</td>
<td>2.6.35 ~</td>
</tr>
</tbody>
</table>
Evaluation environment

• Interconnection of SoCs in embedded systems is similar to cluster connections in that both of them are little packet loss intra-communications.

• Testing SoC’s used for embedded systems
  – (reference [2], evaluation with Xeon, gigabit ethernet)

• Benchmarking throughput as well as CPU load
  – MIPS32®24K® 533MHz × 2, 100Mbps Ethernet
  – Linux Kernel 2.6.20.19, TIPC Version 1.7.7-rc1
Test cases

• (1) Max throughput
  – Send data in one direction.
  – Measure throughput and CPU load per message size

• (2) Latency
  – Data flows in both directions alternately in a ping-pong fashion.
  – Measure the roundtrip time per message size.

• (3) CPU load for Streaming
  – Send data in one direction at a constant rate.
  – Measure the CPU load.
(1) Max throughput, CPU load

- **Message size:** 512, 1024, 1536, ... 40448 Byte
  - Watch a change depending on message size.
  - Compare throughput and CPU load of TIPC with TCP/IP.

- **Measure the time of send()/recv() 10,000 times.**
(1) Max throughput

- TCP/IP < TIPC
  - Header: TCP/IP: 40Byte, TIPC: 24Byte, MTU: 1500Byte
  - Small header $\rightarrow$ high transfer efficiency

![Message size - Throughput graph](image-url)
(1) Max throughput

• TIPC throughput drops in case one more packet needs to be sent.

![Graph showing throughput vs. message size]

- Throughput [Mbps]
- Message size [Byte]

2952 Byte = (1500-24) × 2

packet: 2 → 3
Reason of TIPC throughput drop

TIPC:

SoC A $\rightarrow$ send() $\rightarrow$ SoC B

- 1500Byte $-$ 24Byte = 1476Byte.
- 2953Byte = 1476Byte $\times$ 2 + 1Byte

TCP/IP:

SoC A $\rightarrow$ send() $\rightarrow$ SoC B

- 1500Byte $-$ 40Byte = 1460Byte.
(1) CPU load

- CPU load of TIPC is less than TCP/IP

![Graph showing CPU load vs message size for send and recv operations.]
Packets observance

- **TIPC performance in case of small size packets**
  - Counting RX/TX packets.

```
eth0  Link encap:Ethernet  HWaddr XX:XX:XX:XX:XX:XX
     UP  BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
     RX packets:0  errors:0  dropped:0  overruns:0  frame:0
     TX packets:0  errors:0  dropped:0  overruns:0  carrier:0
     collisions:0  txqueuelen:1000
     RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
```

---

**ifconfig**

```
SoC A  Message  SoC B
```
Message Packets: TCP/IP < TIPC

- Message size: 512 Byte
- TCP/IP small packets coalesce.

![Graph comparing TCP/IP and TIPC](image)

**TCP/IP**

- Number of packets:
  - 0
  - 50
  - 100
  - 150
  - 200
  - 250
  - 300

**TIPC**

- Number of packets:
  - 0
  - 50
  - 100
  - 150
  - 200
  - 250
  - 300

**Send packets**

**Recv packets**
ACK: TCP/IP > TIPC

- Message size: 4096 Byte
- TIPC is less frequent ACK.

<table>
<thead>
<tr>
<th>Number of packets</th>
<th>Time [sec]</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>10000</td>
<td>20</td>
</tr>
<tr>
<td>20000</td>
<td>40</td>
</tr>
<tr>
<td>30000</td>
<td>60</td>
</tr>
<tr>
<td>40000</td>
<td>80</td>
</tr>
</tbody>
</table>

TCP/IP

- Send packets
- Receive packets

TIPC

- Send packets
- Low CPU load

ACK: TCP/IP > TIPC
(2) Latency

• TCP/IP’s latency nearly equals TIPC’s.
(3) CPU load in sending streaming data.

- Data transmission rate: 25Mbps

<table>
<thead>
<tr>
<th></th>
<th>TCP/IP</th>
<th>TIPC</th>
<th>TCP/IP - TIPC</th>
</tr>
</thead>
<tbody>
<tr>
<td>SoC A (send)</td>
<td>9.16</td>
<td>5.18</td>
<td>3.98</td>
</tr>
<tr>
<td>SoC B (recv)</td>
<td>19.25</td>
<td>10.25</td>
<td>9.00</td>
</tr>
</tbody>
</table>
Use TIPC (1/2)

• TIPC doesn’t support SO_SNDFTIMEO.
  → Blocking-send is hard to use.
  → Nonblocking-send repeat.
    → CPU load increases. (test program: ~5%)
Use TIPC (2/2)

• Processes/threads issue a lot of send request → TIPC send queue is filling up.

  (congestion control)
  
  – Increase processes waiting for queue to decrease.
  – Even if queue decreases, and processes get up, wait again without issue a send request.
  – Need to schedule between processes.
Conclusion

- **TIPC is faster than TCP/IP**
  - Smaller header size
  - TCP/IP is faster in case of small message size.
    - Merging small messages would improve the throughput.
- **TIPC has lower CPU load than TCP/IP**
  - Less frequent ACK (other reasons would exist)

- **Verified that TIPC is useful for embedded systems.**
  - (e.g. Reducing CPU load when transferring streaming data.)
Reference

• Reference:

