

## BLIP - A UDP packet's path

Complete code walkthrough for a single packet being sent, forwarded, and received in TinyOS's BLIP stack.

Last updated: Nov 7, 2012

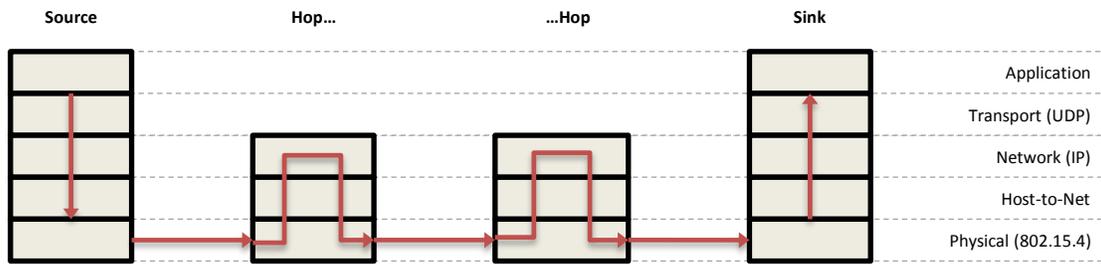


Figure 1 Layers involved in a packet's path through the network

## Mapping from IP stack layering to BLIP code

Quick reference for anyone looking through the code

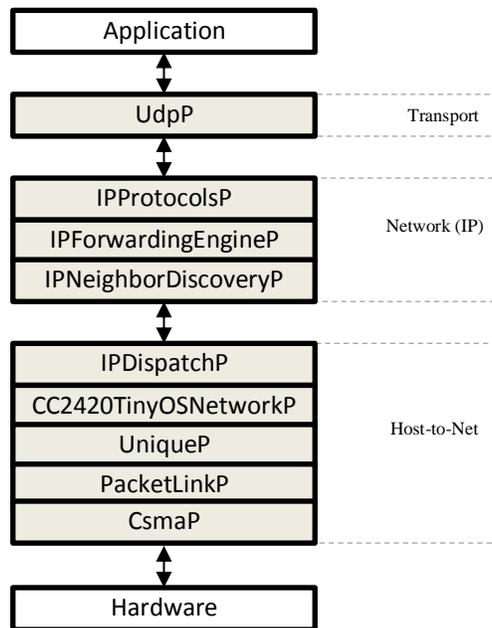


Figure 2 Component Layering in BLIP

## Core concepts

- Static allocation across the stack (no dynamic allocations. Can be a programming pain.)
- Assuming CC2420 radio (just relevant to one part below)
- Assumes basic NesC syntax familiarity

## Function calls

### Packet transmission from source

UDP\_BCP\_BLIPP::Status::sendto

UdpP::UDP::sendto ("/opt/tinyos-2.1.2/tos/lib/net/blip/interfaces/UDP.nc")

UdpP::UDP::sendto "/opt/tinyos-2.1.2/tos/lib/net/blip/UdpP.nc"

- Add IOvector base for payload

UdpP::UDP::sendtov "/opt/tinyos-2.1.2/tos/lib/net/blip/UdpP.nc"

- Add UDP header as IO vector

UdpP::IP::send ""/opt/tinyos-2.1.2/tos/lib/net/blip/interfaces/IP.nc""

IPProtocolsP::IP::send ""/opt/tinyos-2.1.2/tos/lib/net/blip/IPProtocolsP.nc""

- Add TTL, "IP" type

IPProtocolsP::SubIP::send ""/opt/tinyos-2.1.2/tos/lib/net/blip/interfaces/IP.nc""

IPForwardingEngineP::IP::send ""/opt/tinyos-2.1.2/tos/lib/net/blip/IPForwardingEngineP.nc""

- Lookup next hop to reach dest in routing table

IPForwardingEngineP::do\_send "/opt/tinyos-2.1.2/tos/lib/net/blip/IPForwardingEngineP.nc"

- Get some 8byte thing from Pool
- IPForwardingEngineP::IPForward::send
- If send failed, return to Pool

IPForwardingEngineP::IPForward::send[ROUTE\_IFACE\_154] ""/opt/tinyos-2.1.2/tos/lib/net/blip/interfaces/IPForward.nc""

IPNeighborDiscoveryP::IPForward::send "/opt/tinyos-2.1.2/tos/lib/net/blip/IPNeighborDiscoveryP.nc"

- Resolve next hop IPv6 address to LL address (else drop)
- Resolve source IPv6 address to LL address (else drop)

IPNeighborDiscoveryP::IPLower::send "/opt/tinyos-2.1.2/tos/lib/net/blip/interfaces/IPLower.nc"

---

Srikanth Nori

[Autonomous Networks Research Group](#)

University of Southern California

Page 2

IPDispatchP::IPLower::send "/opt/tinyos-2.1.2/tos/lib/net/blip/IPDispatchP.nc"

- Check if free frag available
  - o If not, ERETRY
- Clear fragment (IPDispatchP::BarePacket::clear)
  - o IPDispatchP::BarePacket::clear "/opt/tinyos-2.1.2/tos/interfaces/Packet.nc"
  - o CC2420TinyosNetworkP::BarePacket::clear "/opt/tinyos-2.1.2/tos/chips/cc2420/lowpan/sim/CC2420TinyosNetworkP.nc"
- Pack IOVectors into compressed 802.15.4 format in a byte-buffer
  - o IPDispatchP::lowpan\_frag\_get "/opt/tinyos-2.1.2/support/sdk/c/blip/lib6lowpan/lib6lowpan\_frag.c"
    - IPDispatchP::pack\_ieee154\_header
    - IPDispatchP::lowpan\_pack\_headers -
    - IPDispatchP::pack\_nhc\_chain
    - ...
- IPDispatchP::BarePacket::setPayloadLength - Finalize packet
- IPDispatchP::SendQueue::enqueue - Enqueue into send queue
- IPDispatchP::PacketLink::setRetries - Set retries
- Repeat for all fragments
- IPDispatchP::sendTask::postTask – prepare to send
  - o IPDispatchP::sendTask
  - o

Separate task to send out packets IPDispatchP::sendTask::runTask "/opt/tinyos-2.1.2/tos/lib/net/blip/IPDispatchP.nc"

- Get entry from send queue
- IPDispatchP::ieee154Send::send "/opt/tinyos-2.1.2/tos/interfaces/Send.nc"
- CC2420TinyosNetworkP::BareSend::send "/opt/tinyos-2.1.2/tos/chips/cc2420/lowpan/sim/CC2420TinyosNetworkP.nc"
- CC2420TinyosNetworkP::SubSend::send "/opt/tinyos-2.1.2/tos/interfaces/Send.nc"
- UniqueSendP::SubSend::send "/opt/tinyos-2.1.2/tos/interfaces/Send.nc"
  - o Check if unique ("dsn")
- PacketLinkP::Send::send
  - o Until success/num retries
    - PacketLinkP::SubSend::send "/opt/tinyos-2.1.2/tos/interfaces/Send.nc"
    - CC2420CsmaP::Send::send "/opt/tinyos-2.1.2/tos/chips/cc2420/csma/sim/CC2420CsmaP.nc"
  - o From this point on it reaches hardware

## Packet reception at destination

### Event signaled from hardware interfaces

CC2420CsmcP::Receive::receive "/opt/tinyos-2.1.2/tos/interfaces/Receive.nc"

- Check if this packet is for me (imitating CC2420)

UniqueReceiveP::SubReceive::receive "/opt/tinyos-2.1.2/tos/chips/cc2420/unique/sim/UniqueReceiveP.nc"

- Ensure this is not a duplicate. If it's been seen before then drop here silently

UniqueReceiveP::Receive::receive "/opt/tinyos-2.1.2/tos/interfaces/Receive.nc"

CC2420TinyosNetworkP::SubReceive::receive "/opt/tinyos-2.1.2/tos/chips/cc2420/lowpan/sim/CC2420TinyosNetworkP.nc"

- Check if this is "Bare" packet
- AMs are routed elsewhere, not relevant to BLIP. BLIP needs low-level access.

CC2420TinyosNetworkP::BareReceive::receive "/opt/tinyos-2.1.2/tos/interfaces/Receive.nc"

IPDispatchP::leee154Receive::receive "/opt/tinyos-2.1.2/tos/lib/net/blip/IPDispatchP.nc"

- Unpack header from the byte-buffer
- If header indicates fragmented packet
  - o Check if this is start . If it is then start a fresh recon buffer
  - o If it is middle then add to prev recon buffer
  - o If it is end then finalize recon buffer
- If not fragmented
  - o Insert into recon buffer
- In both cases call next event below with recon buffer as input

IPDispatchP::deliver "/opt/tinyos-2.1.2/tos/lib/net/blip/IPDispatchP.nc"

- Get the recon buffer
- Free it after below call

IPNeighborDiscoveryP::IPLower::recv "/opt/tinyos-2.1.2/tos/lib/net/blip/IPNeighborDiscoveryP.nc"

IPNeighborDiscoveryP::IPForward::recv "/opt/tinyos-2.1.2/tos/lib/net/blip/interfaces/IPForward.nc"

IPForwardingEngineP::IPForward::recv ""/opt/tinyos-2.1.2/tos/lib/net/blip/IPForwardingEngineP.nc""

- Signal raw IP packet, if anyone is interested
- Check if it is destined for local address

- If not, forward to next hop (see next section)
- else

IPForwardingEngineP::IP::recv "/opt/tinyos-2.1.2/tos/lib/net/blip/interfaces/IP.nc"

IPProtocolsP::SubIP::recv "/opt/tinyos-2.1.2/tos/lib/net/blip/IPProtocolsP.nc"

- Get IP header
- Check that it is valid IP packet header

IPProtocolsP::IP::recv "/opt/tinyos-2.1.2/tos/lib/net/blip/interfaces/IP.nc"

- Check the protocol type UDP/TCP/ICMP

UdpP::IP::recv "/opt/tinyos-2.1.2/tos/lib/net/blip/UdpP.nc"

- Check the port if someone has called "bind()" for this port
- Check UDP checksum

UdpP::UDP::recvfrom[my "instance" of UDP] "/opt/tinyos-2.1.2/tos/lib/net/blip/interfaces/UDP.nc"

UDP\_BCP\_BLIPP::Echo::recvfrom "UDP\_BCP\_BLIPP.nc"

### Packet forwarding at intermediate hop

IPForwardingEngineP::IPForward::recv ""/opt/tinyos-2.1.2/tos/lib/net/blip/IPForwardingEngineP.nc""

- Signal raw IP packet, if anyone is interested
- Check if it is destined for local address
- If it is NOT, then we need to fwd this to the next hop
- Get IP header from IO vector
- Lookup Route to next hop from fwding table

IPForwardingEngineP::do\_send "/opt/tinyos-2.1.2/tos/lib/net/blip/IPForwardingEngineP.nc"

- Continue as before. No change.