Anatomy of Routers and Switches

Petr Grygárek

using a couple of slides
of Martin Stankuš
(with his permission)

Routers

Router Operating Systems

Router Operating System

- General-purpose OS
- Specialized OS
  - minimum OS overhead
    - many safeguards omitted
  - non-preemptive
  - interrupt-based
- As the complexity of control plane grows, today's modern router OS tends to become modular
  - SW fault isolation, memory-effectiveness, …

Components of typical router OS

- Kernel
- Device drivers
- Processes
  - System maintenance
  - User interface
  - Routing Protocols
- Packet Switching
- Packet Buffers

Memory organization

- Flat virtual address space
- No paging/swapping
- Divided into multiple regions
  - SRAM for packet buffers
    - Fast access
  - DRAM for OS image, OS and process's data structures
    - including running configuration
- Memory managed using memory pools

Processes

- No memory protection using virtual memory
  - No memory management during context switches
  - Every processes can access whole memory
- Non-preemptive (run-to-completion model)
  - Lower overhead
  - Lower programming complexity
    - synchronization issues, deadlocks
- Created/terminated on startup or by other process
  - e.g. by user interface
- Static process priorities
  - used to plan another process when previous one relinquishes control
  - process with lower priority can run only when there is no process with the higher priority

Kernel

- Manages system resources
  - Schedules processes
  - Manages memory
  - Provides interrupt handlers
  - Maintains timers
- Does not run in processor mode different from mode of other processes
  - as general-purpose OS kernels obviously do

Scheduler

- Maintains process queues
  - Ready queue
    - Processes eligible to run
    - Separate queue for every priority level
  - Idle queue
    - Active processes waiting on an event to occur before they can continue to run
  - Dead queue
    - Processes that have terminated
    - Process resources have to be reclaimed before total removal from the system
- Watchdog timer

Memory manager

- Multiple memory pools
- Block allocation, deallocation + refragmentation
- Manages data memory used by processes and packet buffers

Packet buffer management

- Needed for store-and-forward packet routing
- Must process packets of various sizes
  - Multiple chains of buffers of various sizes
  - Static preallocation or dynamic allocation
    - parameters: min buffers, max allowed
  - System performance tuned by preallocating appropriate number of buffers of particular size
    - depends on characteristics of traffic passing the router

Device drivers

- Provides abstract view to hardware devices
  - Control
  - Data
- Interface with rest of system defined using data structures containing entry points for individual driver's functions
- Drivers for FLASH, NVRAM, network interfaces, ...

Packet switching

- Not computation-intensive, but data-intensive
- Basic technique: switching of every packet by software using routing table longest match
  - Platform-independent
  - Possibility of per-packet load sharing
- Software optimization: various types of route cache
  - Only per-source/per-destination load sharing
- Hardware support on some platforms

Switching by software process

- Network interface hardware receives packet and copies it into receive memory (using DMA)
- Network interface interrupts CPU to inform it about new packet
- CPU places the packet to the input queue of appropriate switching process (e.g. IP) and makes this routing process eligible to run
- After switching process is planned to run, it
  - finds the next hop router
    - Using routing table (possibly recursive search)
  - Constructs frame to encapsulate forwarded packet
    - Using ARP table
  - Places packet into the outbound interface queue
    - QoS mechanisms may be implemented here
- Outbound interface hardware dequeues packet and sends it out
  - After that it informs CPU by interrupt that it can free the queue buffer

Caching of routing results

- First packet routed by switching process
- Resulting information is cached
  - Destination IP+prefix length, outbound interface, outbound frame MAC header
  - Cache implemented by hash tables or radix trees (256-way)
- Other packets to the same destination matching cache entry bypass switching process completely
  - Entire packet switching operation completed during interrupt
- Effective for traffic flowing to limited number of destinations
  - Not effective for Internet backbone
- When routing table change occurs, cache has to be deleted

Router hardware architectures

Router architecture history

- Software-based routers
  - shared memory, 1 CPU, 1 bus
- Separation of control plane and data path
  - multuple buses (CPU bus, line cards bus)
  - Proprietary, PCI
- Distribution of intelligence to multiple line cards
  - full-mesh P2P lines between line cards and route processor
- Multi-chassis distributed switching/routing fabric

Specialized architectures

- Network processor with instructions optimized for packet switching
- Multiple buses introduced to ensure fast data transfer
- Short time to transfer packets from interface memory to CPU main memory and back
- Distributed forwarding:
  - Forwarding information base (FIB) copied to individual interface cards
    - FIB is created by main processor based on routing table
      - Recursive lookups resolved
    - Interface card equipped with route processor is able to route between it's ports without interruption of main processor

Switches (L2)

Control Plane Functions

- Address learning
  - if not implemented by HW
- Spanning Tree
- Port bundling protocols
- Management
- ...

Components of typical switch (1)

- Backplane
  - high-performance bus (older approach)
  - multicast/broadcast implementation easy (no replication)
  - commonly oversubscribed
  - access arbiter
- Switching fabric
  - full mesh of P2P lines between line cards (new approach)
  - sometimes hierarchical
    - local on line modules + central
    - CAM/FIB table replication and distributed maintenance needed

Components of typical switch (2)

- Port modules (line cards)
- Contents-associative memory (CAM)
  - Bridging table allows parallel search
  - Search/maintenance commonly implemented by ASIC

Throughput of switches

- Aggregated-throughput
- Nonblocking switch

Frame buffering

- Even for cut-through switching, frame has to be buffered if
  - Multiple frames are to be sent out from the same interface at the same time
  - Speeds of input and output ports differ
- Most often buffers of maximum frame size are allocated (1514B)
- Multiple queues per port needed if QoS is required
  - fixed size queues to reach deterministic delay

Frame buffering options

- Buffers at input ports
  - Frame which cannot be sent right now may block processing of input queue
- Buffers at output ports
  - Possibly multiple output queues for various traffic classes (QoS support)
  - Most-common implementation option
- Shared memory
  - Most space-effective
    - Not needed to allocate fixed number of buffers for every input/output port as in previous options
    - Buffers allocated on as-needed basis
  - Need to synchronize memory access from input and output port modules

Circuitry for SOHO Switches

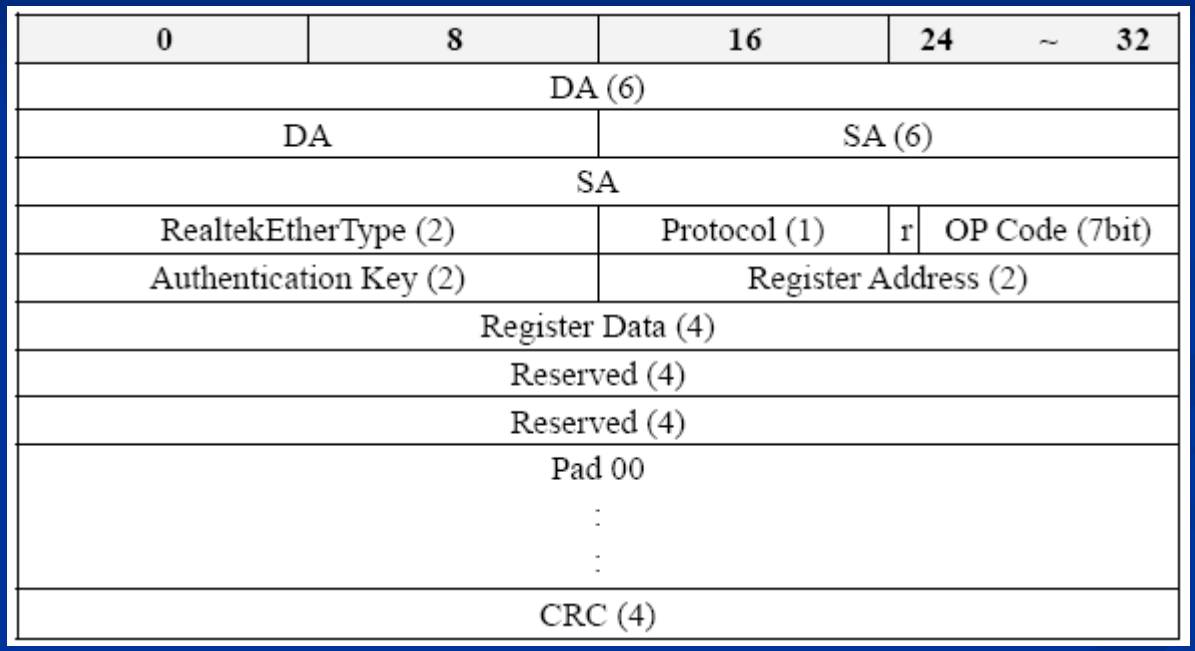
Example: ASIC Realtek RTL8316BP

- 16 x 100Base-TX / 100Base-T MAC
- Store-and-forward architecture
- 8k MAC Lookup table
- 160 kB packet buffer
- Auto MDI / MDIX
- 802.3x flow control
- Configuration options:
  - I2C (Master+EEPROM, Slave+CPU)
  - „Pin Strapping“
  - RRCP
- HW IGMP Snooping
- Port Mirroring
- Per-port bandwidth control
- Per-port event counters
- 32 VLANs
  - port based
  - 802.1Q based
- QoS
  - port based
  - 802.1p based
  - IPv4 TOS/DSCP based
- Weighted round robin
  - (1:4, 1:8, 1:16, high priority first)

RRCP: Realtek Remote Control Protocol

- Proprietary L2 protocol
- Publicly available documentation
- Plain text autehentication
- GET_REGISTER, SET_REGISTER, GET_REPLY, HELLO, HELLO_REPLY
- Open source implementation OpenRRCP for Linux

RRCP Frame



0	8	16	24 ~ 32
DA (6)			
DA		SA (6)	
SA			
RealtekEtherType (2)		Protocol (1)	r OP Code (7bit)
Authentication Key (2)		Register Address (2)	
Register Data (4)			
Reserved (4)			
Reserved (4)			
Pad 00 : :			
CRC (4)			

Block diagram