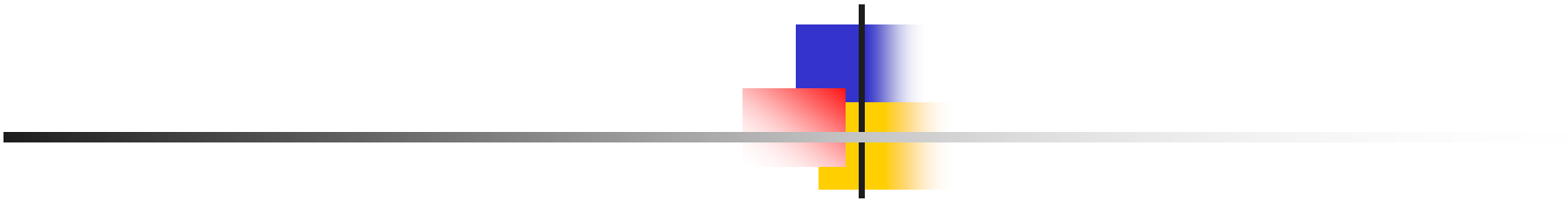
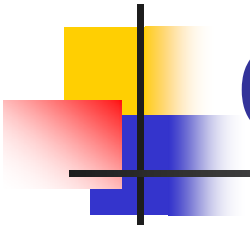
# CSE398: Network Systems Design



Instructor: Dr. Liang Cheng
Department of Computer Science and Engineering
P.C. Rossin College of Engineering & Applied Science
Lehigh University

April 11, 2005

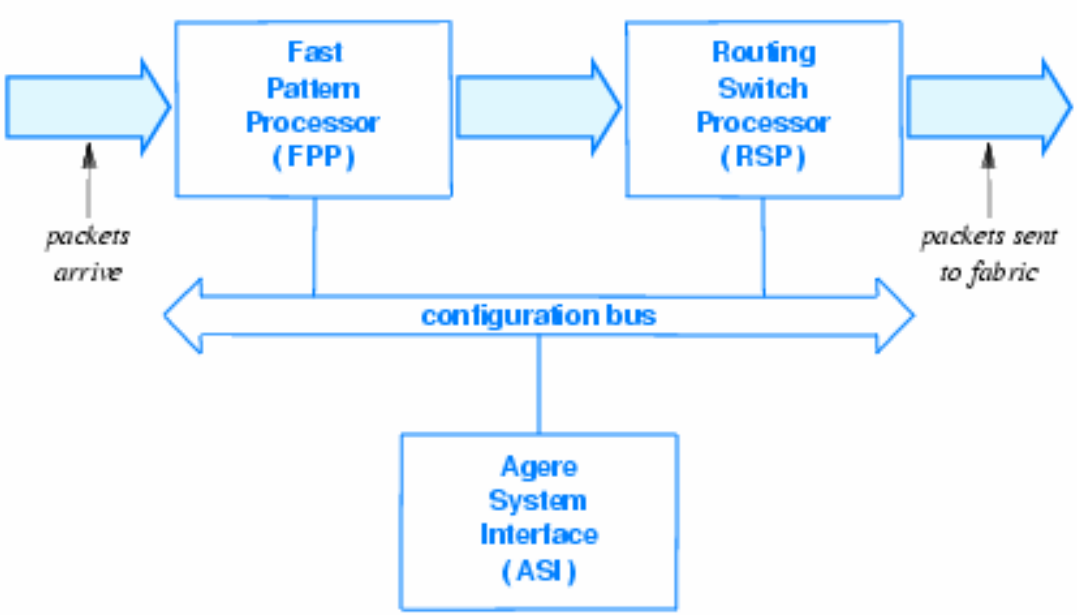


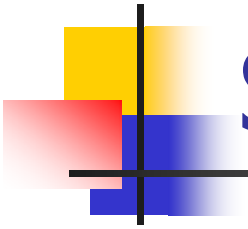
# Outline

- Recap
  - Reminder: homework due today
  - Overview of Agere's network processor
- State engine and scripting language
- Summary

# First Generation before Agere PayloadPlus (APP)

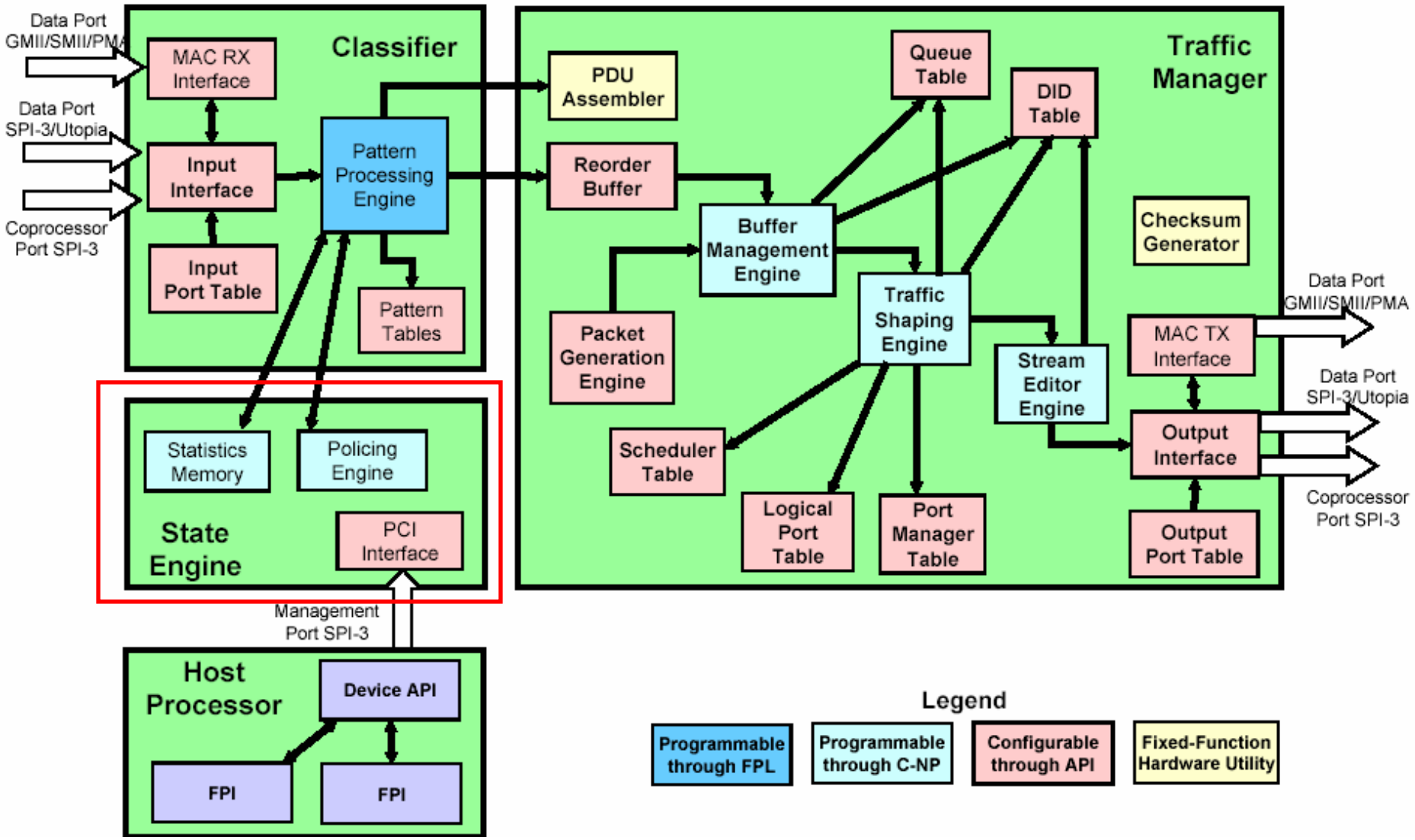
- Three separate chips
- FPP+RSP: fast path of data plane
- ASI: statistics, interface to a host microprocessor for system mgmt.



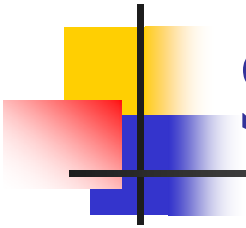


# Second Generation

- A single IC but various models
  - APP550: four GigE ports with full capability
    - Classification: pattern processor
    - Forwarding: traffic manager and modifier
    - Statistics and host communication: state engine


Data Port
GMII/SMII/PMA
Data Port
SPI-3/Utopia
Coprocessor
Port SPI-3
MAC RX
Interface
Input
Interface
Input
Port Table
Classifier
Pattern
Processing
Engine
Pattern
Tables
PDU
Assembler
Reorder
Buffer
Queue
Table
DID
Table
Traffic
Manager
Buffer
Management
Engine
Checksum
Generator
Packet
Generation
Engine
Traffic
Shaping
Engine
Stream
Editor
Engine
MAC TX
Interface
Data Port
GMII/SMII/PMA
Data Port
SPI-3/Utopia
Coprocessor
Port SPI-3
Scheduler
Table
Logical
Port
Table
Port
Manager
Table
Output
Interface
Output
Port Table
Statistics
Memory
Policing
Engine
State
Engine
PCI
Interface
Management
Port SPI-3
Host
Processor
Device API
FPI
FPI
Legend
Programmable
through FPL
Programmable
through C-NP
Configurable
through API
Fixed-Function
Hardware Utility

# State Engine (SE)

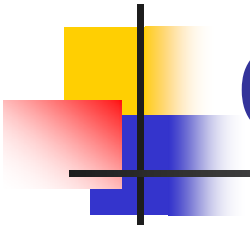


- Contains memory used to store statistics
  - Classifier invokes a SE function to store or increment a value in memory
- Performs computations needed for traffic policing
  - Only the 1st step ?
- Provides an interface to an external host processor
  - Configuring and initializing APP550 - FCRAM
  - Slow path

# External Host Interface

- PCI
  - Peripheral component interconnect
  - Synchronized with the clock speed of the microprocessor
  - Transmits 32 bits at a time in a 124-pin connection and 64 bits in a 188-pin connection in an expanded implementation
  - Sending the address on one clock cycle and data on the next
  - Burst data: address on the first cycle and a sequence of data transmissions on a certain number of successive cycles

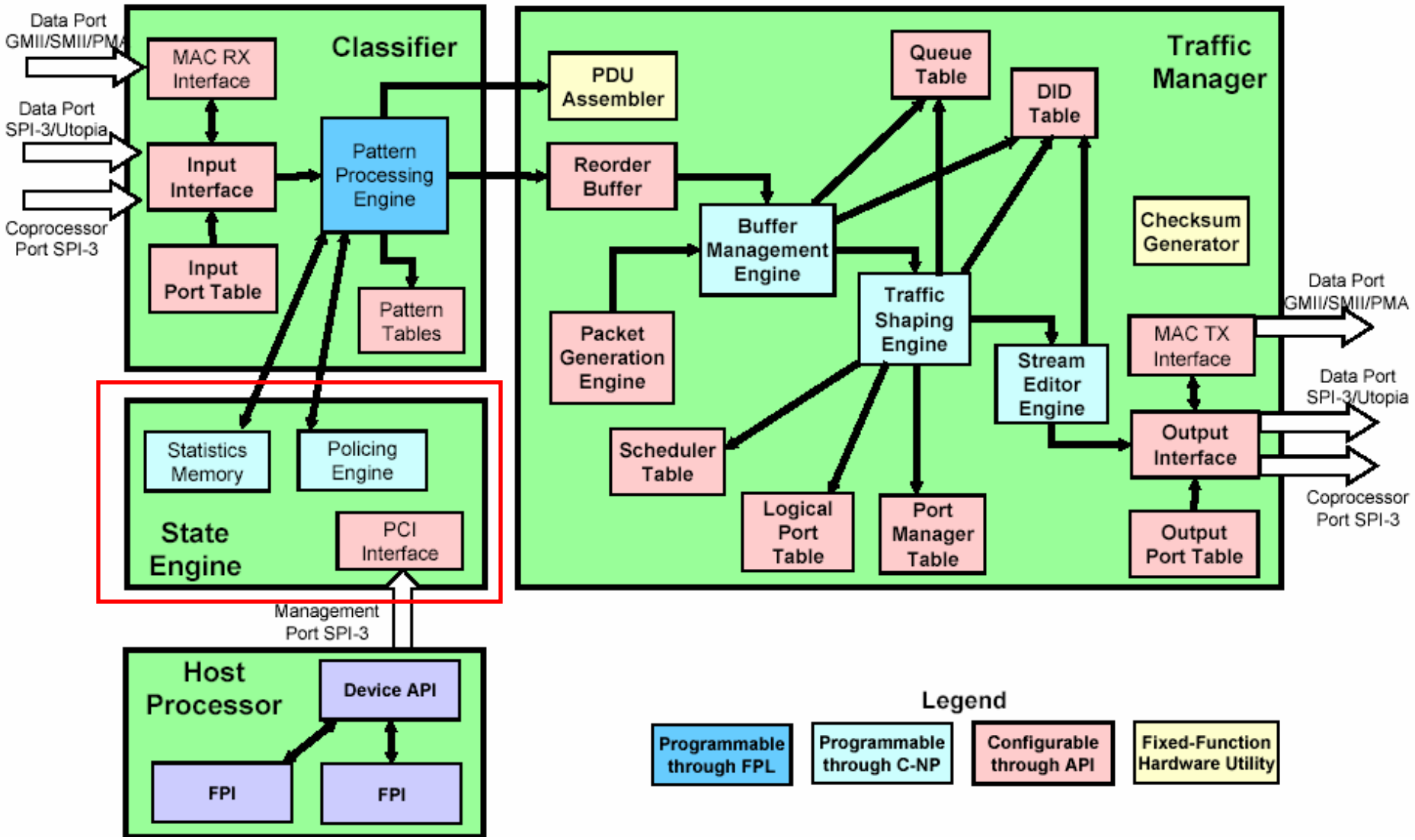
# Control and Status Registers



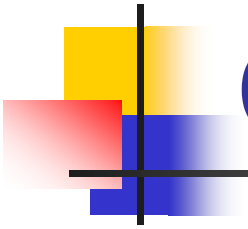
- CSRs
  - Store-and-fetch
  - To control a function unit, place a value in one of the registers
  - To obtain status from a function unit, fetch the value from the register
    - State engine
    - Classifier
    - Traffic manager
    - Internal memory
    - MAC interfaces

# State Engine Memory

- Up to 32MB of external DDR-SRAM
- Up to 2.6MB of internal memory
- 4-byte units: registers
  - 135 registers for base addresses
    - Heads and tails of lists
    - Control and status functions
  - 1280 registers for storing policing scripts
    - Policing script memory (1024) + extended script memory (256)
    - Each instruction is 136 bit long: 128 + 8


Data Port
GMII/SMII/PMA
Data Port
SPI-3/Utopia
Coprocessor
Port SPI-3
MAC RX
Interface
Input
Interface
Input
Port Table
Classifier
Pattern
Processing
Engine
Pattern
Tables
PDU
Assembler
Reorder
Buffer
Queue
Table
DID
Table
Traffic
Manager
Buffer
Management
Engine
Checksum
Generator
Packet
Generation
Engine
Traffic
Shaping
Engine
Stream
Editor
Engine
MAC TX
Interface
Data Port
GMII/SMII/PMA
Data Port
SPI-3/Utopia
Coprocessor
Port SPI-3
Scheduler
Table
Logical
Port
Table
Port
Manager
Table
Output
Interface
Output
Port Table
Statistics
Memory
Policing
Engine
State
Engine
PCI
Interface
Management
Port SPI-3
Host
Processor
Device API
FPI
FPI
Legend
Programmable
through FPL
Programmable
through C-NP
Configurable
through API
Fixed-Function
Hardware Utility

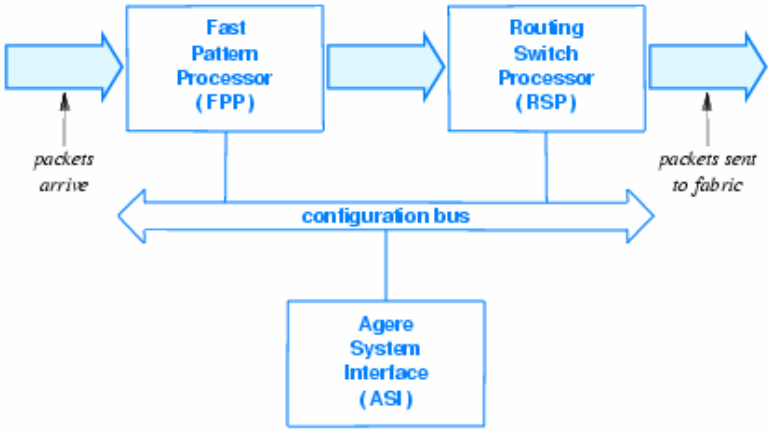
# Onboard Configuration Bus



- CBI (Configuration Bus Interconnect)
  - Connecting Classifier, Traffic manager, and State Engine
  - Control plane functions
  - Implements the address space containing CSRs and SE memory
- CBI - State Engine - PCI

# ASI Functions

- A FPL program uses ASI functions to invoke the SE
  - asiWrite(0x4:24,949:32)
  - Arithmetic and logical operations
    - Read
    - Increment/decrement



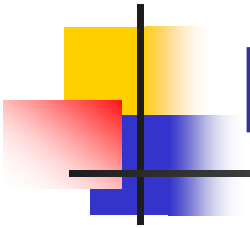
# Policing Engine and Scripts

- A flow in the profile?
  - A real-time clock
  - Policing database
    - Memory holding flow information
    - Indexed by flow ID
    - Up to 512K entries
    - Each entry is of 64 bytes
  - A scripting facility
    - Store and fetch flow statistics to the database
    - Up to 16 separate policing scripts to be invoked by an FPL program during packet classification
      - asiPolice3 or asiPoliceEOF3
    - Each script is compiled to .aso file and loaded to APP550

# Return Values from Policing

- asiPolice does not return a value
- asiPoliceEOF returns a value
  - asiPolice on early blocks, and asiPoliceEOF on the last block
  - For frame traffic, asiPoliceEOF is used in the 2nd pass of an FPL program
  - Minimizes calls to asiPoliceEOF

# Binding a Script to its ID

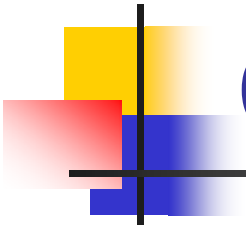


- asiPolice3 or asiPoliceEOF3
  - Using SPA to change the configuration file
  - Edit the XML file
  - <ASIScript file="myscript.aso" id="0" />

# Script Protocol Declaration

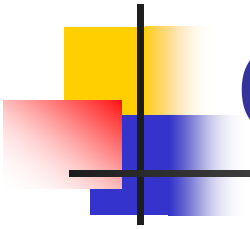
- SETUP PROTO(asiPoliceEOF3, 24, 16, 24)
- Example on Page 321

```
#include “np5.fpl”
#include “mp5asi.fpl”
…
SETUP PROTO(asiPoliceEOF3, 24, 16, 24);
…
outcome = asiPoliceEOF3($FID:24,$currLength:16, 0:24);
… //place outcome in tm_flags
fTransmit(0:1,0:1,$DID:20,0:16,0:5,$tm_flags:10,$info:24);
```



# C-NP

- @[x] *type* : reference the xth byte
- @[x:y] *type* : reference a string of bytes from byte x to byte y
- block entire_packet @[0:63];
- block ip_header packet_data[14:33];
- unsigned srcIP ip_header[12:15];
- unsigned dstIP ip_header[16:19];



# C-NP (cont'd)

- Statements
  - Assignment
  - Condition: if (expression) statement
    else statement
  - Selection: switch (expression)
    case statements
  - Compound: { statement; … statement; }

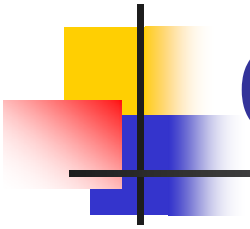
# C-NP (cont'd)

- Scripting structure (an example on pp. 327)

optional preprocessor directives (e.g. #include, #ifdef, …)

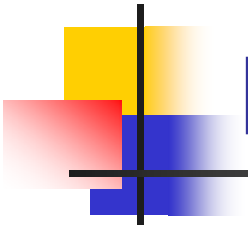
data declaration

**script** script_name {
    script body
}



# Outline

- Recap
- Examples
- Design tradeoffs and consequences
- Summary and homework



# Review Question

- Page 266 Exercise 17.3: What is the advantage of separating classification from forwarding?