

U t ü ä á
b ñ x ü t à ç z
f ç ä x Å
f ä ù v ä ä ù x

B.Visscher

Aug 2001



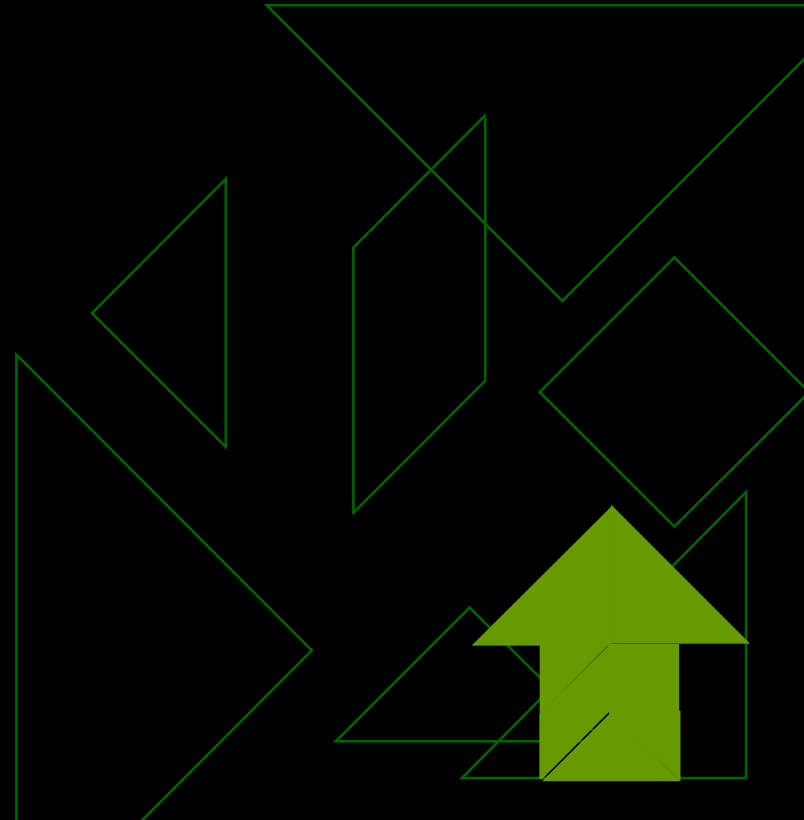
Introduction

- ◆ Design Goals
- ◆ What is BOSS
- ◆ Hardware
- ◆ Dependency Flow Model
- ◆ Implementation resources BOSS
- ◆ Perceptron application
- ◆ Conclusion
- ◆ Questions

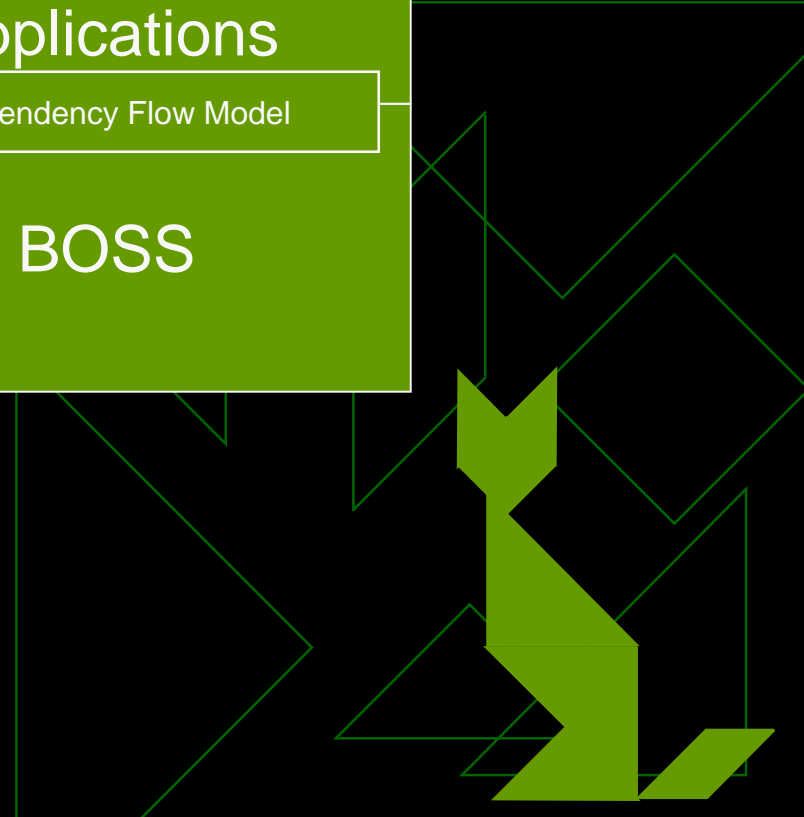
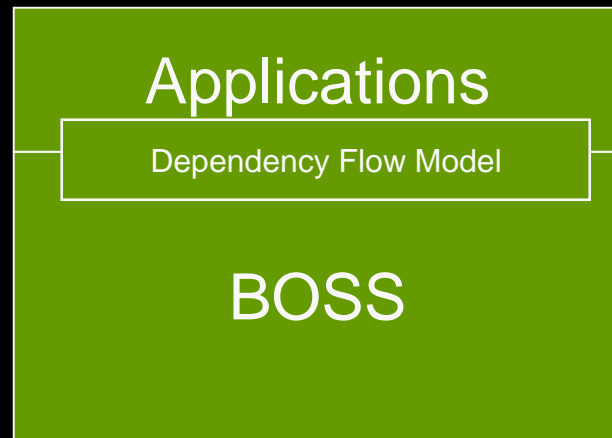
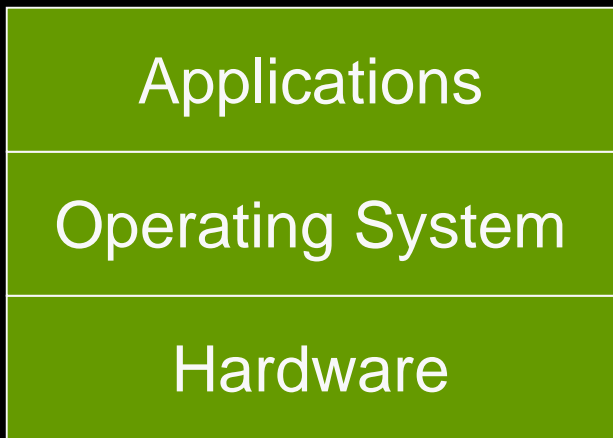
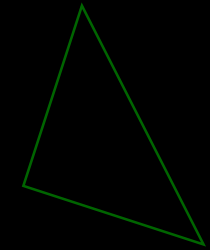


Design Goals

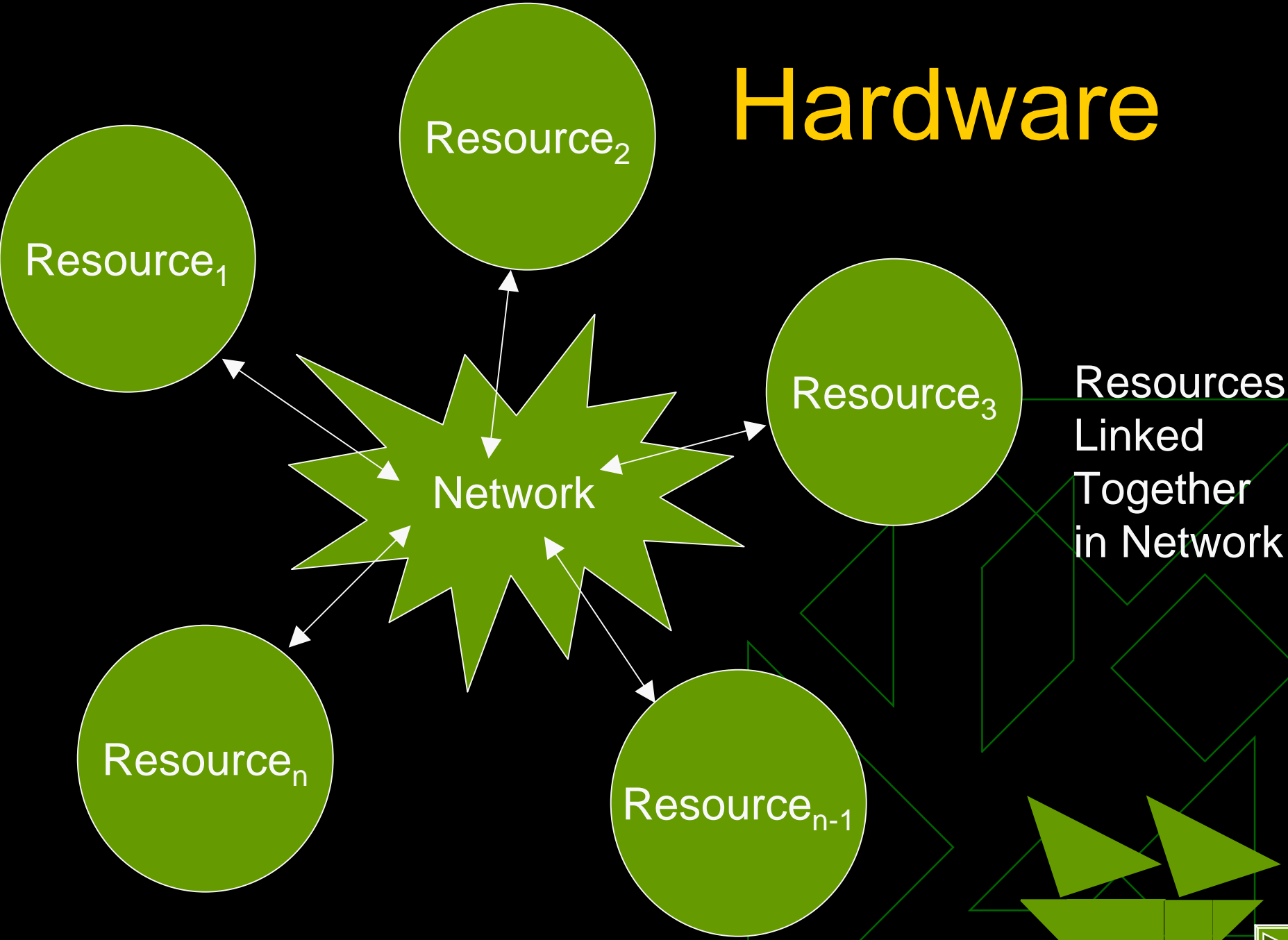
- ◆ Easy to understand and program
- ◆ Transparent Hardware / Software
- ◆ Fast
- ◆ Reliable
- ◆ Safe
- ◆ Free



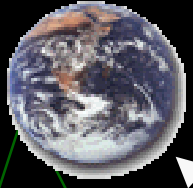
Traditional v.s. BOSS



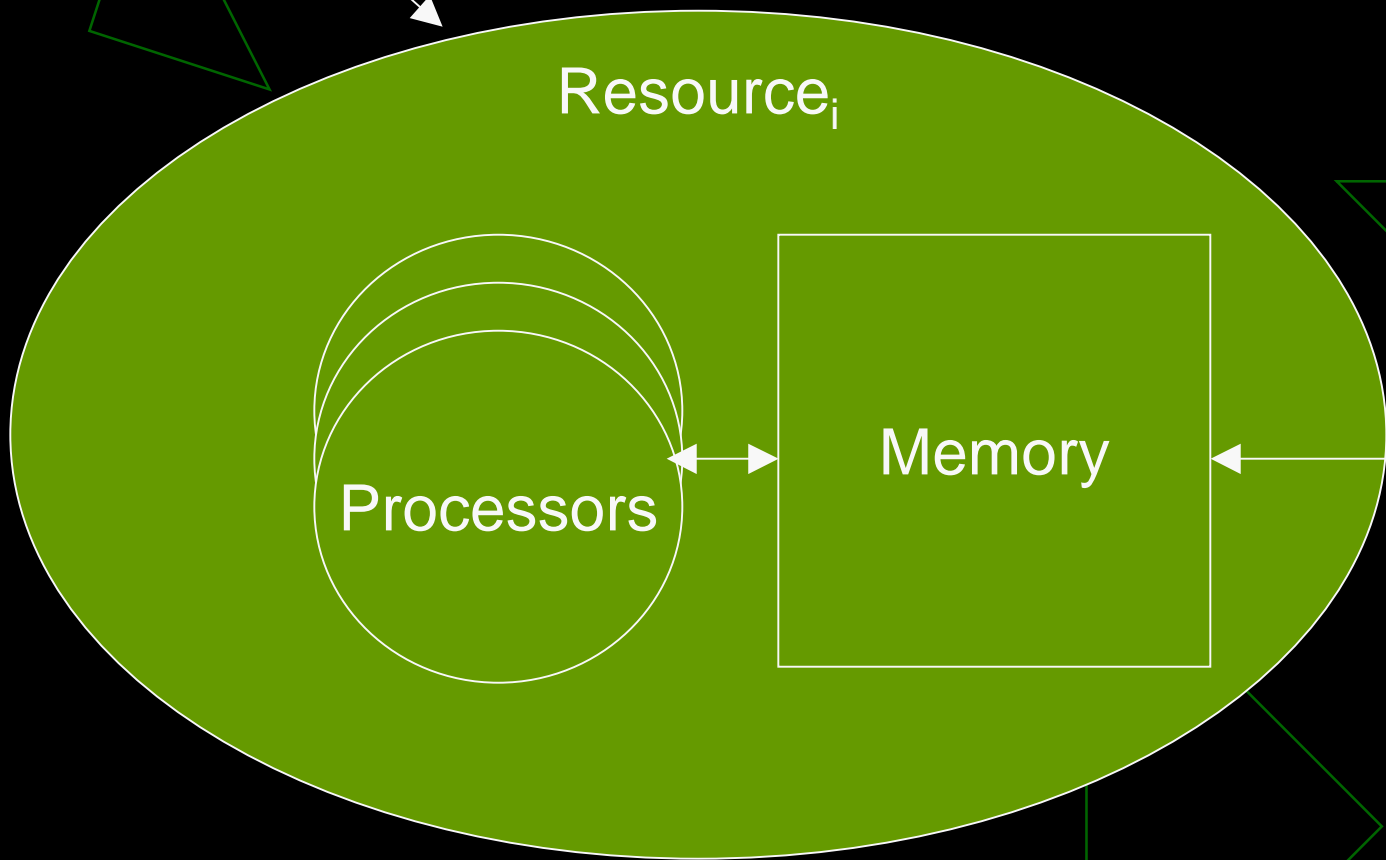
Hardware



Resource



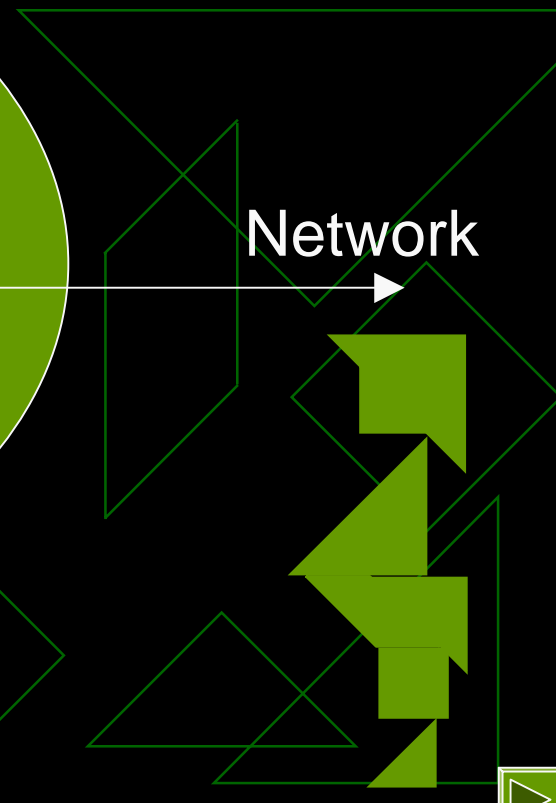
Environment / Side Effects



Resource_i

Processors

Memory



Network

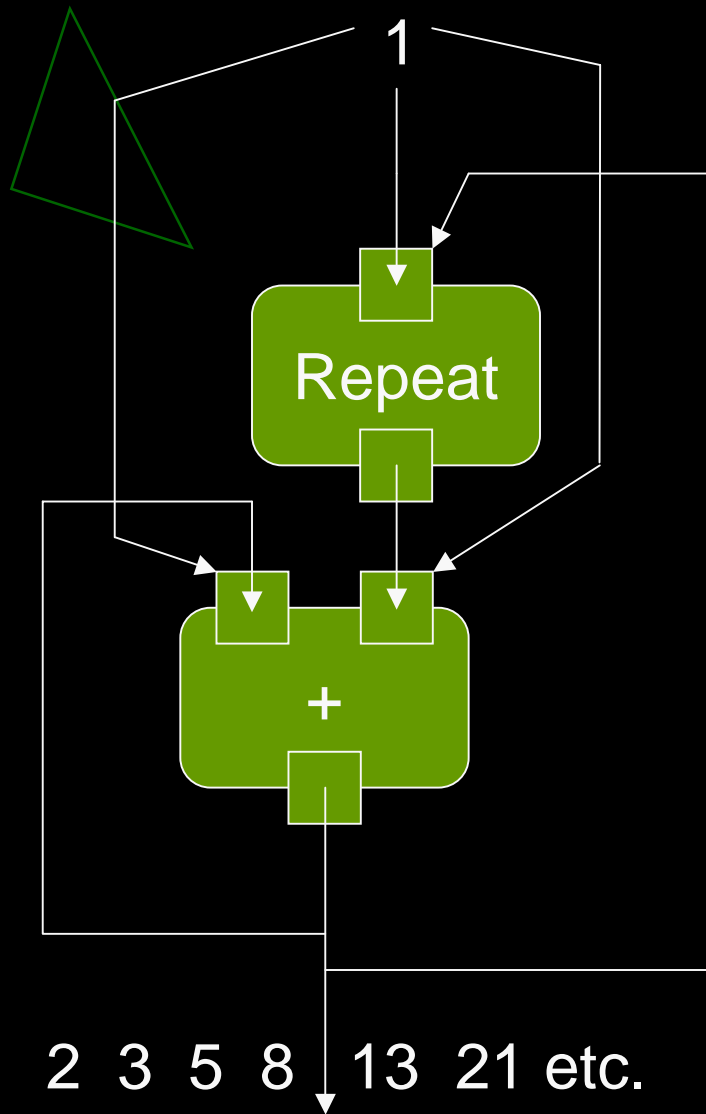
Dependency Flow Model

- ◆ Dependency Flow Networks
 - Programming
- ◆ Owner structure
 - Abstraction
 - Process management
- ◆ Resource Structure
 - Scheduling
 - Information
 - Security



Dependency Flow Network

Programming



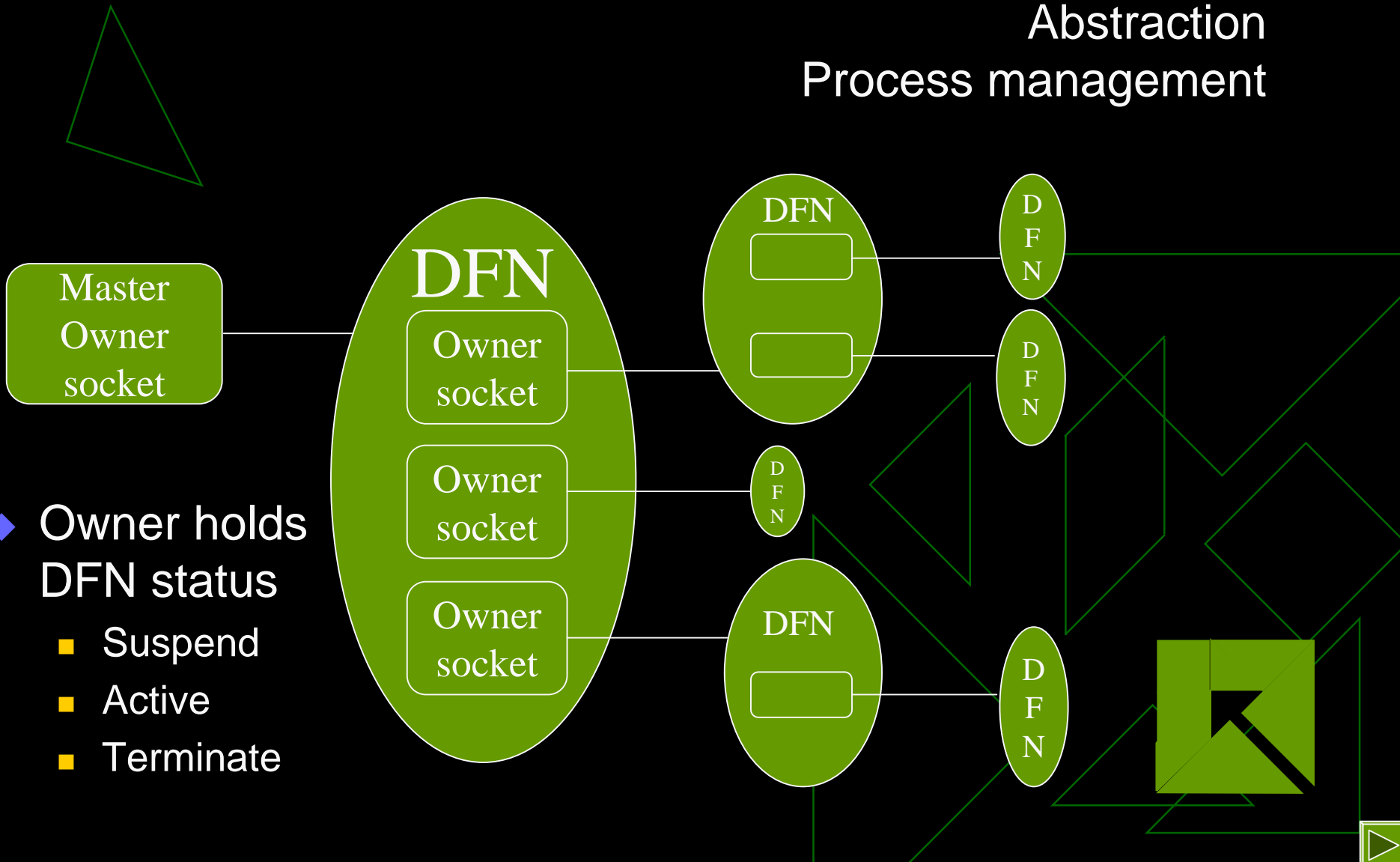
- ◆ Sockets
 - Process
- ◆ Ports
 - Storage
- ◆ Channels
 - Communication



Owner structure

Abstraction

Process management

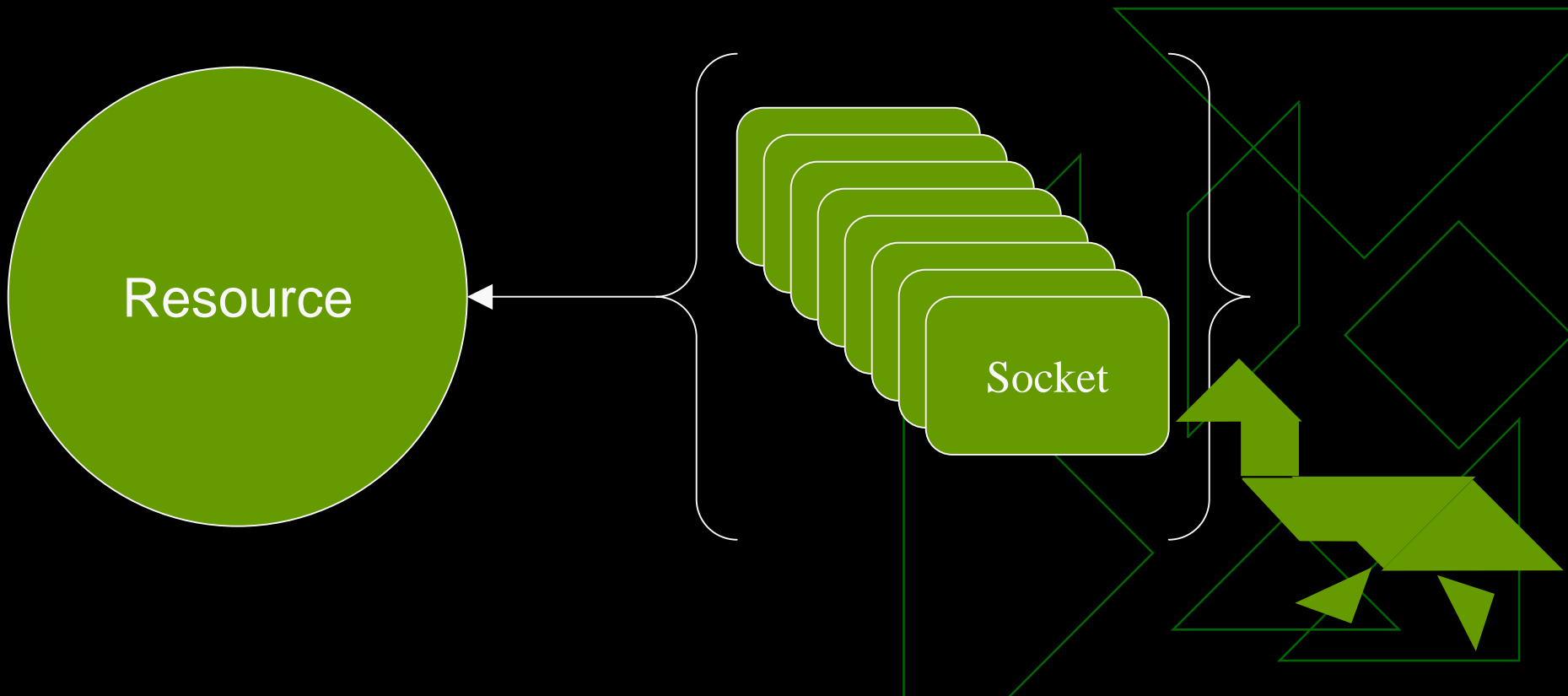


- ◆ Owner holds DFN status
 - Suspend
 - Active
 - Terminate

Resource structure

- ◆ Socket : request for a process
- ◆ Resource : provider of processes

Scheduling
Information
Security




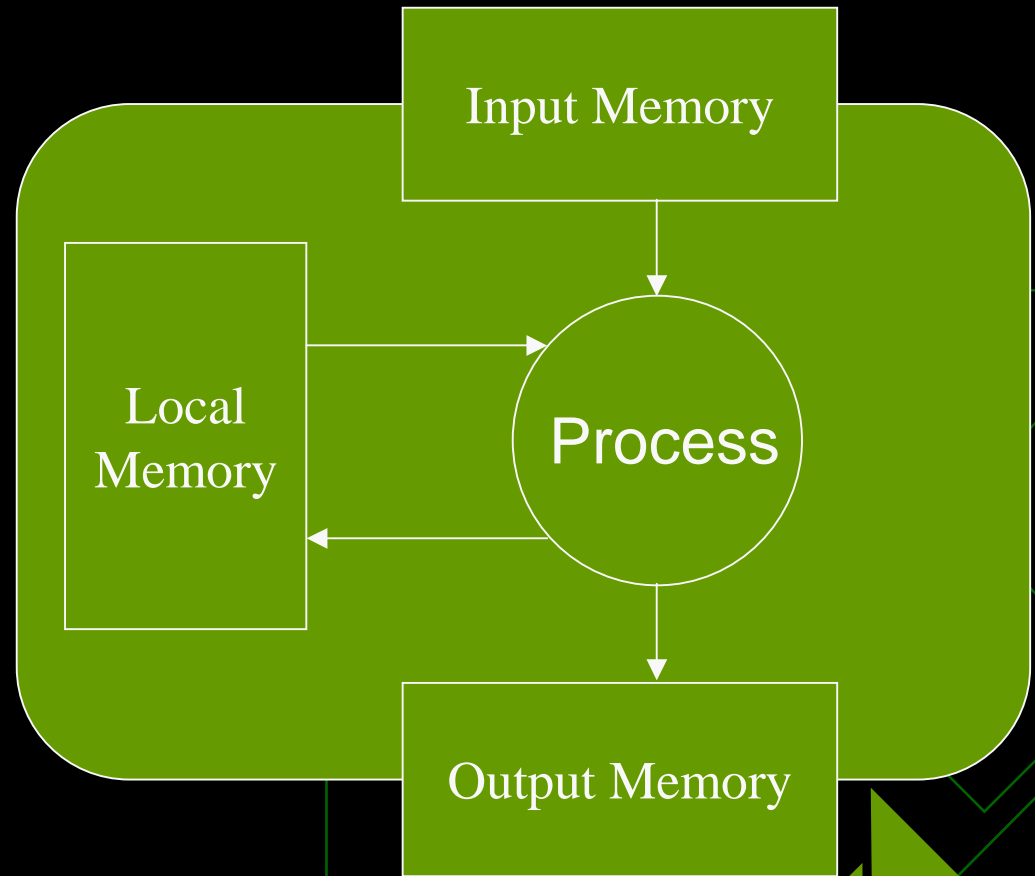
Socket

- ◆ Part of

- DFN 
- Owner structure
- Resource structure

- ◆ Represents

- Resource  process (atomic)
- DFN (combined)



Abstraction

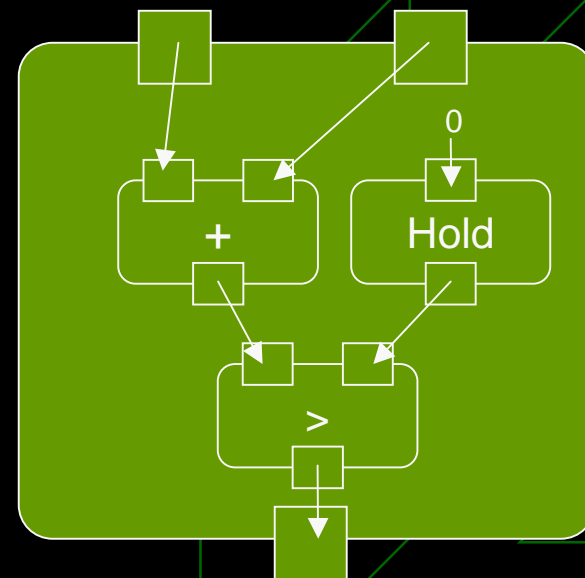
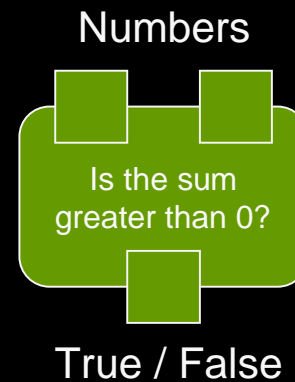
Subroutine
Function

- ◆ Black Box = Owner

- Behavioral description
- Implicit

- ◆ White Box = DFN

- Dependency description
- Explicit

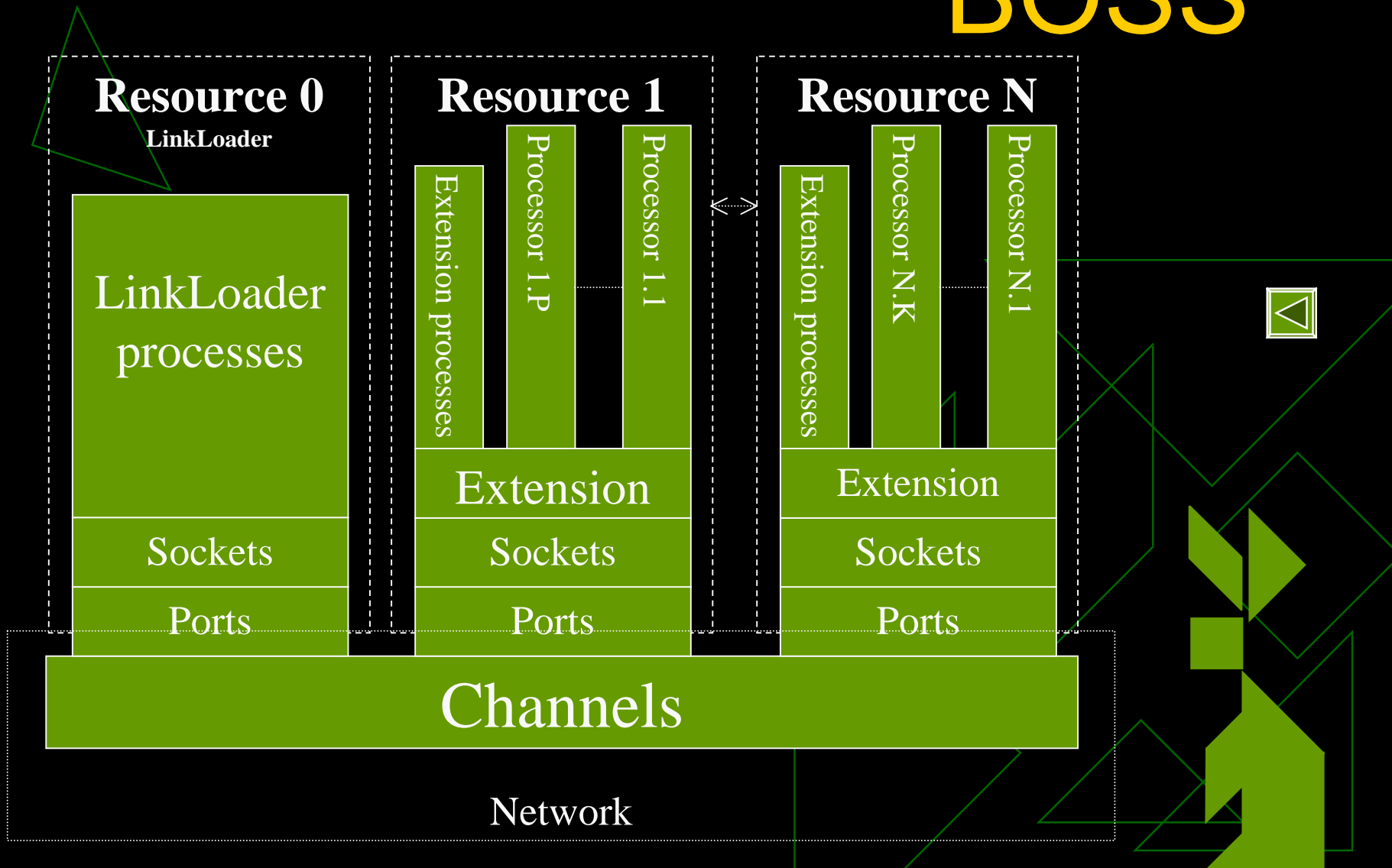


LinkLoader Resource

- ◆ Converts passive dependency data into working processes (DFN)
 - Manages Black box processes
 - ◆ Communicates variables
 - ◆ Introduces constants
 - ◆ Suspends / Activates / Terminates processes
 - Manages Namespaces
 - ◆ Primary namespace
 - ◆ Secondary namespace
 - ◆ Global namespace



Complete Structure: BOSS



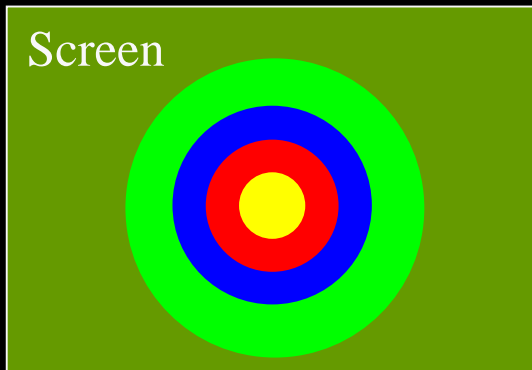
Implemented processes

- ◆ Naming of resource processes
 - `<resource>.<process>(in0,in1...inn)(out0,out1...outn)`
- ◆ Resource: signal, boolean, byte, integer, float, double
 - Example: `boolean.and(boolean,boolean)(boolean)`
 - Example: `integer.+(integer,integer)(integer)`
- ◆ Resource: flow
 - Processes: repeat, hold, sync, switch, merge, last, after
 - All processes for data types : signal,boolean,...,@,NIL)
- ◆ Resource: text user interface, TUI

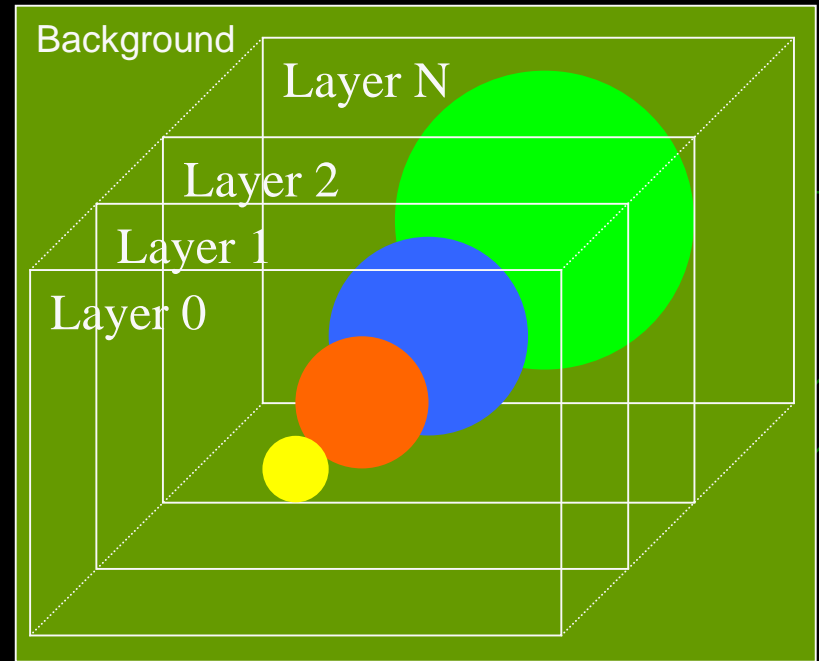


TUI Resource

Demo



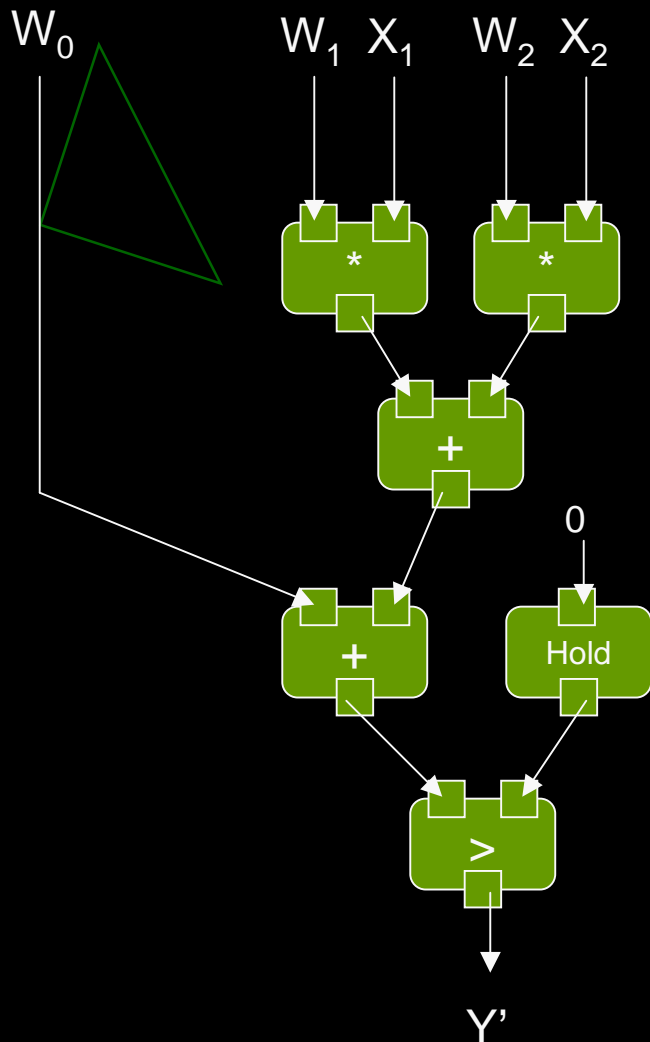
← Layer
Holder



- ◆ Screen divided into multiple layers
- ◆ Background color

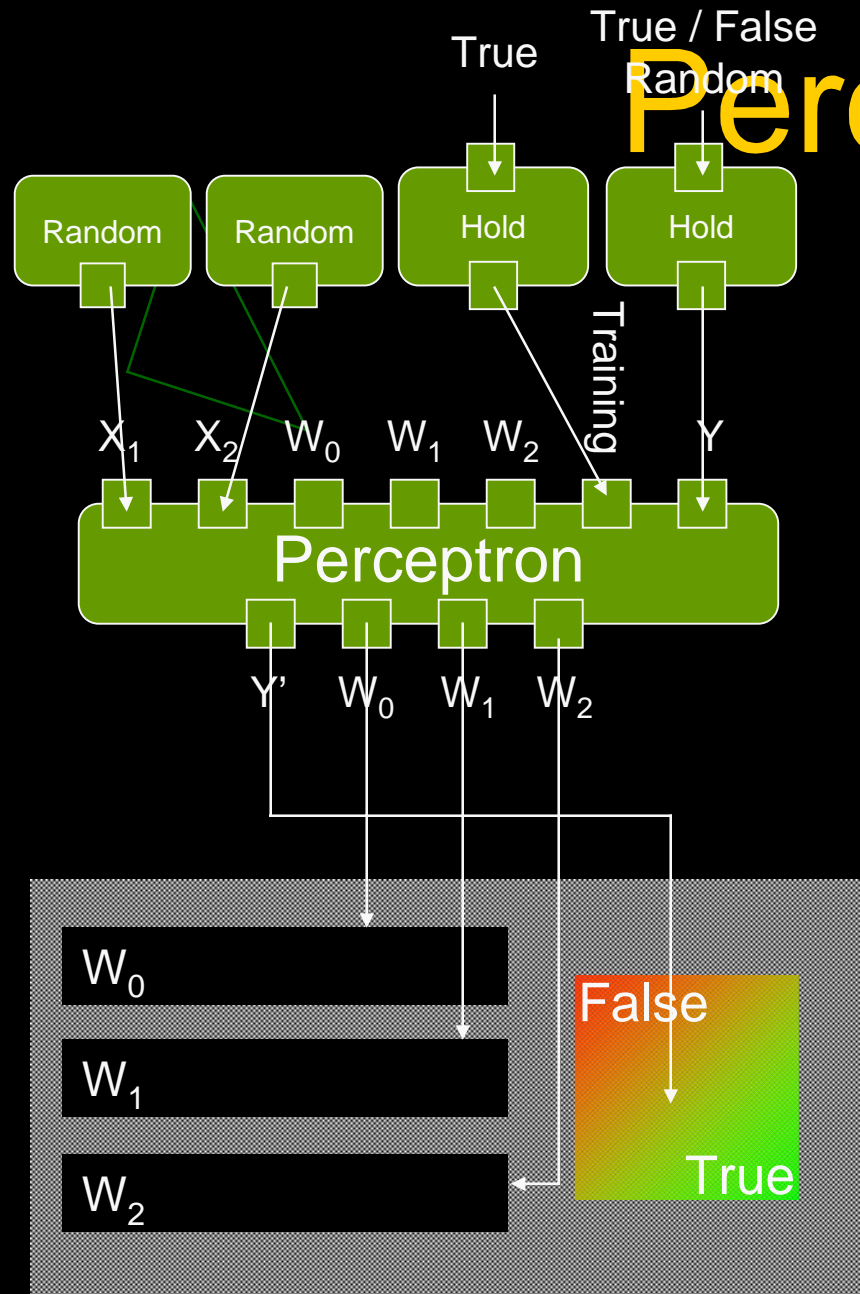


Perceptron

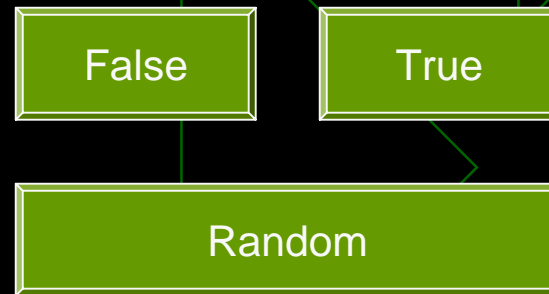


- ◆ Running
 - $Y' = (W_0 + W_1 * X_1 + W_2 * X_2) > 0$
- ◆ Training (Determining W)
 - $Y = \text{True}, Y' = \text{False} \rightarrow (Y - Y') = 1$
 - $Y = \text{False}, Y' = \text{True} \rightarrow (Y - Y') = -1$
 - $W_i^{\text{new}} = W_i^{\text{old}} + (Y - Y') * X_i$
- ◆ $Y, Y' : \text{boolean } \{\text{True}, \text{False}\}$
- ◆ $X, W : \text{integer } \{-n \dots -2, -1, 0, 1, 2 \dots n\}$

Perceptron Demos



- ◆ Random Value range: $[-127, 127]$
- ◆ $W_0, W_1, W_2: 1$
- ◆ Training: True
- ◆ Y : True, False, Random



Conclusion

- ◆ Easy to understand and program / Free
 - DFN
 - Owner structure / Abstraction
 - Information in resources
- ◆ Transparent Hardware / Software
 - Atomic processes
 - Combined processes
- ◆ Fast
 - Parallel execution of processes
- ◆ Reliable / Safe
 - Resource structure
 - Autonomy of resources

