Example: Simple Router
endif
forward packet out port X
decrement addr;
else
send packet out port Y
remove addr from packet;
if addr == 0, then
receive a packet;
loop forever

\* At each receiving port A and B, do: 

Simple Router - FUNCTIONAL DESCRIPTION
concurrent, any edges into Sink assumed concurrent.

Except, any edges from Source node (graph root) assumed

progress (node). Edges represent communication dependencies.

communication action is considered a separate sequential

In deriving the communication flow graph, initially each named

( [ i | i ] ; ? B ) * || ( [ i | i ] ; ? A ) *

Omit operational details, abstract communication actions.

Simple Router - COMMUNICATION BEHAVIOR
Predecessors, sink is ignored when checking for successors.

and end of graph. Source is ignored when checking for start.

Source and sink are special vertices, represent start/stop conditions.

Edge attributes represent additional sequencing constraints.

Represented by vertex v.

An edge e represents communication action.

A set of edges. An edge e? means communication action.

Represented by vertex v? proceeds from vertex v.

A set of vertices. Each vertex or node represents a

Directed graph G = (V, E), with

Communication Flow Graph
Communication Flow Graph

\[(A?; [X! | Y!]) \parallel (B?; [X! | Y!])\]
Graph Topologies for Mapping

Nodes with IN Degree = 1

IN Degree = 1

OUT Degree = 1

Router

OUT Degree = 2

Buffer

OUT Degree = 2

Arbiter

IN Degree = 2

OUT Degree = 1

Mutex
Group Nodes by Edges Out
3x3 Router

```latex
3x3 Router

\begin{align*}
\text{SRC} & \rightarrow \text{Xi} \\
\text{Xi} & \rightarrow \text{Vi} \\
\text{Vi} & \rightarrow \text{Pout} \\
\text{Pout} & \rightarrow \text{Pin} \\
\text{Pin} & \rightarrow \text{B?} \\
\text{B?} & \rightarrow \text{A?} \\
\text{A?} & \rightarrow \text{Sink} \\
\end{align*}

\text{\textbf{SRC}} \rightarrow \text{Xi} \rightarrow \text{Vi} \rightarrow \text{Pout} \rightarrow \text{Pin} \rightarrow \text{B?} \rightarrow \text{A?} \rightarrow \text{Sink}
3x3 Router

\begin{align*}
Pout : & \text{tristable mutex} \\
Y : & \text{tristable mutex} \\
X : & \text{2 input mutex}
\end{align*}

\begin{align*}
* ( A? ; [ X! | Y! | Pout! ] ) & \quad || \\
* ( B? ; [ Y! | Pout! ] ) & \quad || \\
* ( Pin? ; [ X! | Y! | Pout! ] )
\end{align*}
3x3 Router - Another Grouping

A? (APin? ; (X! | Pout! | Y!))
| |
|| (B? ; (Pout! | Y!))

Group by Out Edges

APin : Arbiter
Group by in Edges

3x3 Router - Another Grouping - cont’d

(((in \ Pout) \ Y) \ B) * || (((in \ Y) \ X) \ A) *