

A Fast and Energy-Efficient Stack

Jo Ebergen, Daniel Finchelstein (MIT),
Russell Kao, Jon Lexau, David Hopkins

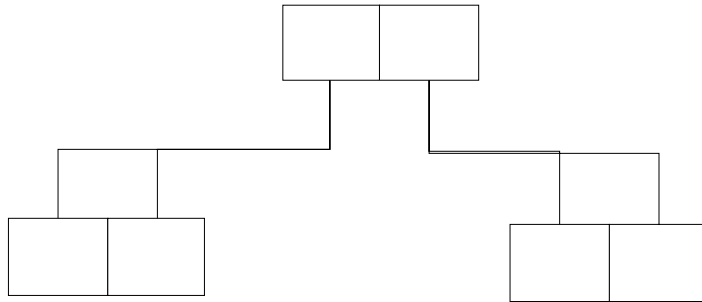
Asynchronous Design Group
Sun Microsystems Laboratories, CA

Talk Outline

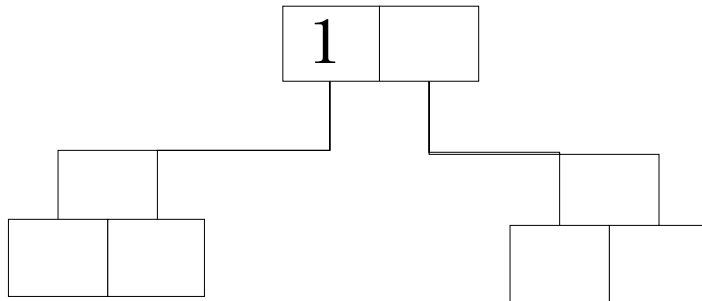
- **Linear and tree stacks**
- **Stack FSM implementation**
- **FSM to GasP conversion**
- **Strategy for testing the stack**
- **Chip results**

- **Last-In-First-Out (LIFO) data structures**
- **Applications: function call/return, expression evaluation, linked lists**
- **Two commands: PUT (push) and GET (pop)**
- **Functionality, Cycle Time, and Power Consumption**

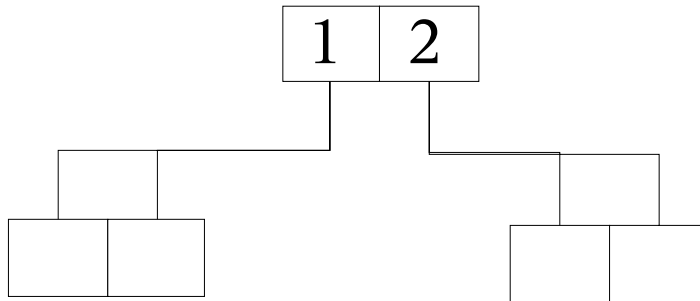
Tree Stack Example



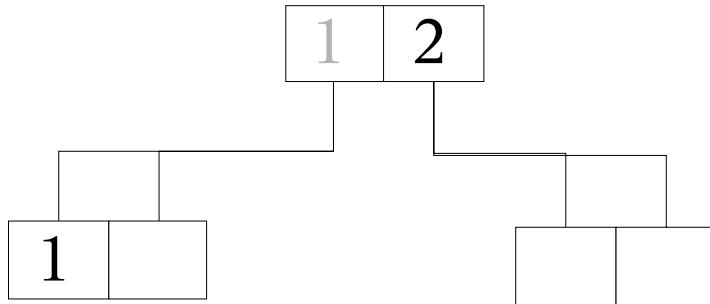
PUT 1



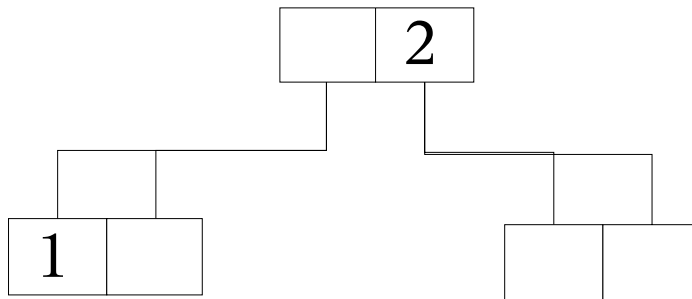
PUT 2



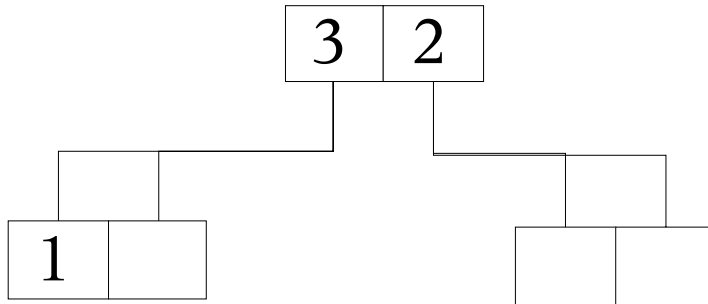
PUT 2



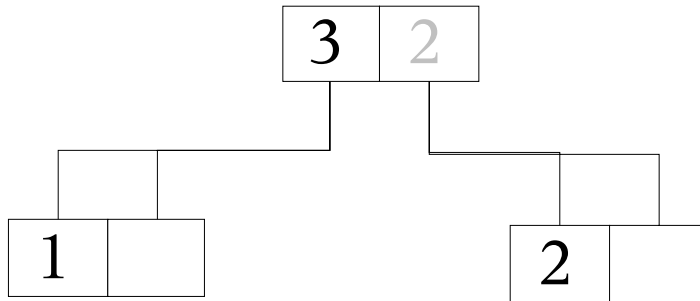
PUT 3



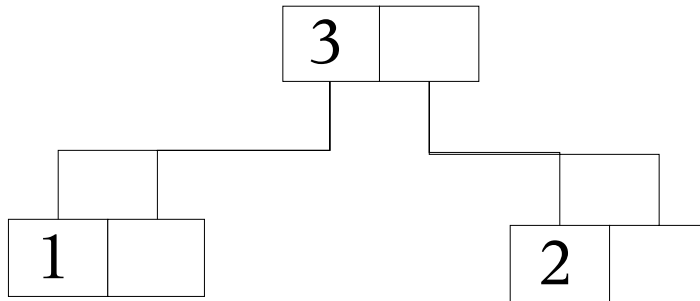
PUT 3



PUT 3

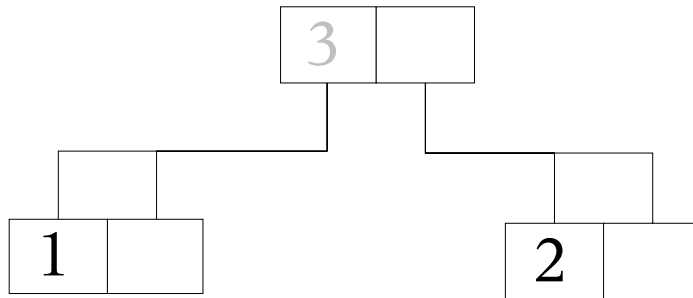


GET (3)

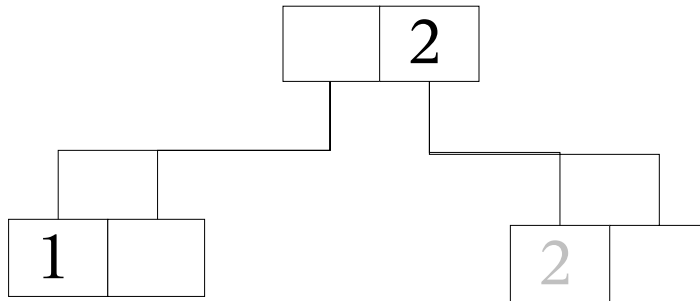


GET (3)

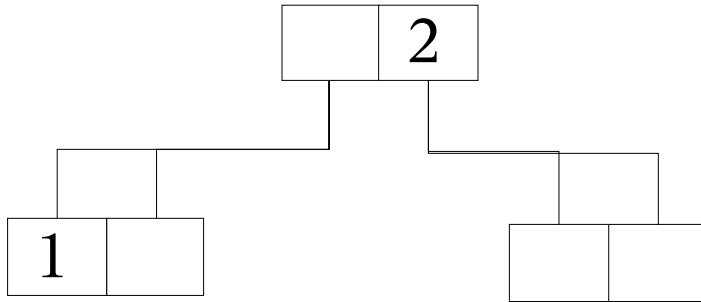
3



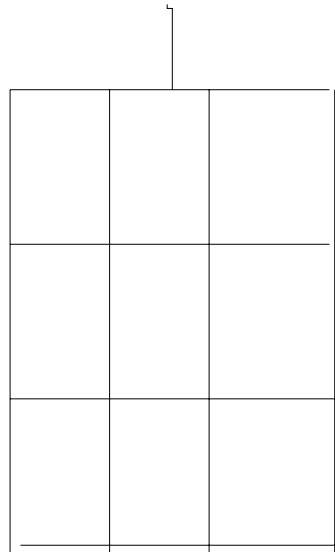
GET (3)



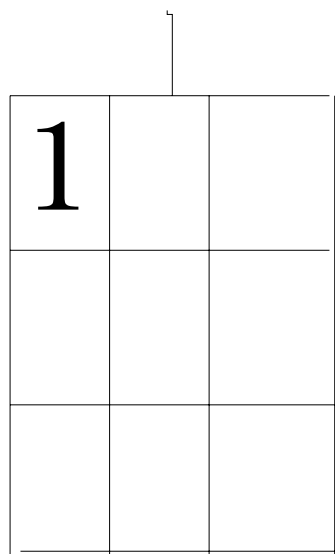
GET (3)



Linear Stack Example



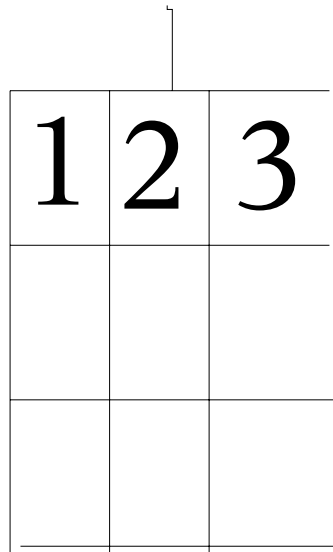
PUT 1



PUT 2

1	2	

PUT 3



PUT 3

1	2	3
1		

PUT 3

	2	3
1		

PUT 4

4	2	3
1		

PUT 4

4	2	3
1	2	

PUT 4

4		3
1	2	

GET (4)

4

4		3
1	2	

GET (4)

		3
1	2	

GET (3)

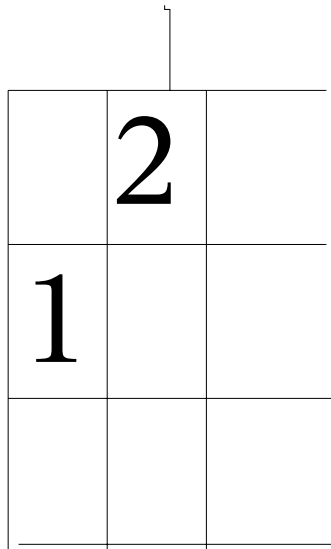
3

		3
1	2	

GET (3)

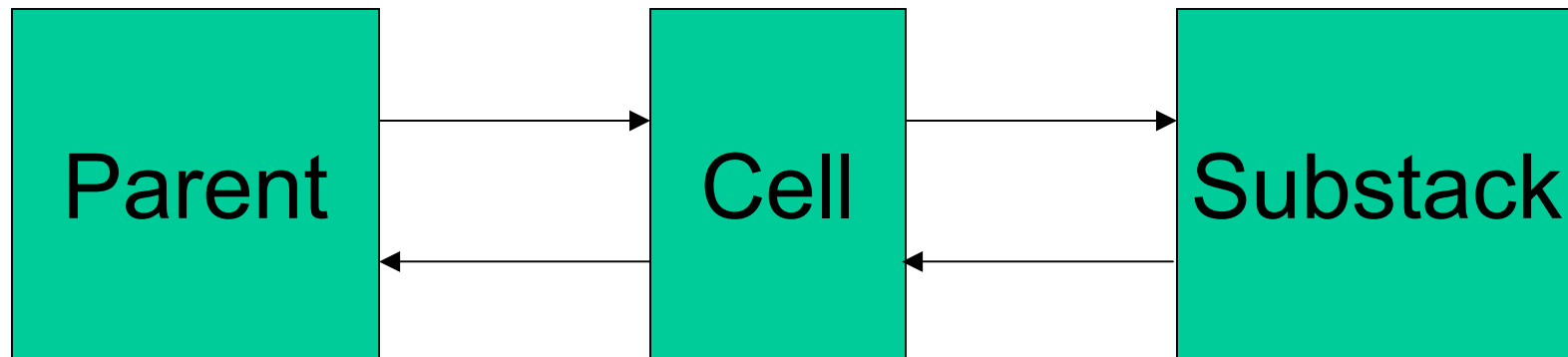
	2	
1	2	

GET (3)



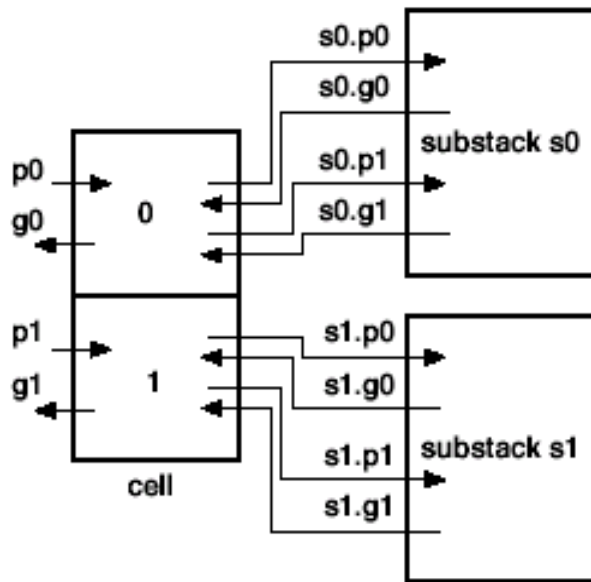
Stack Cell

- Complete stack = composition of stack cells
- Cell interfaces with parent and substack (see figure)
- Datapath: pulse latches controlled by put and get actions
- Control implemented by FSM in GasP

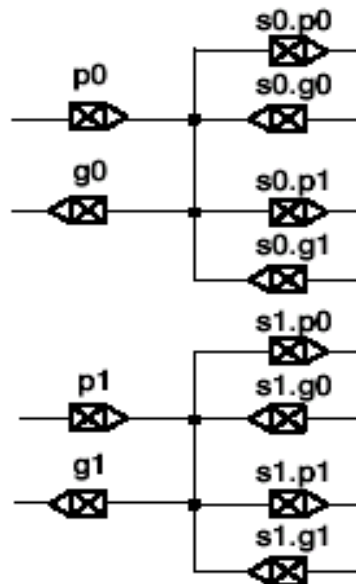


Datapath Movements

2-Place Tree Stack



Cell Actions

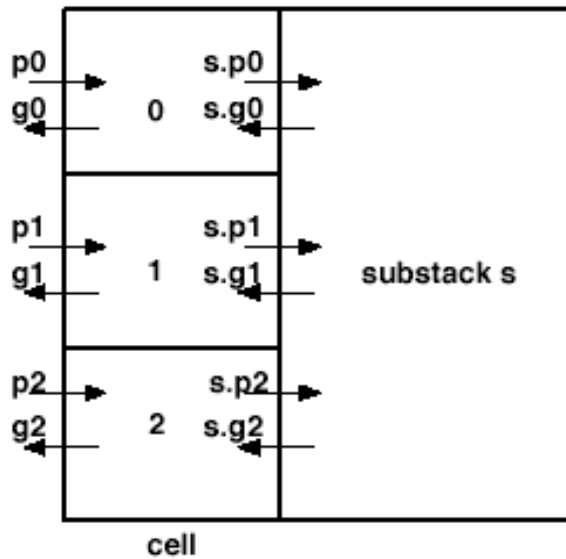


Cell Datapath

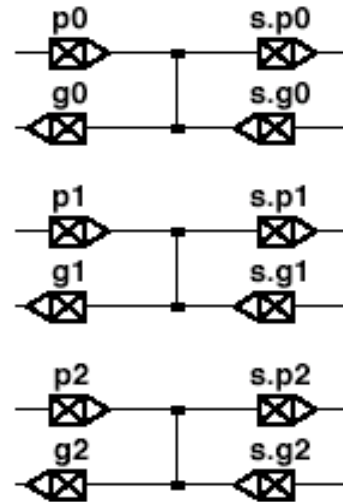
```
state E where
  E = ( p0 -> N1
        | gU -> E )
  ..
  N1 = ( p1 -> P0
         | g0 -> G1 )
  ..
  P0 = ( s0.p0 -> N0
         | s0.p1 -> N0
         | s0.pU -> F )
  ..
end
```

FSM Snippet

3-Place Linear Stack



Cell Actions



Cell Datapath

```

state E where
  E = ( p0 -> N1
      | gU -> E )

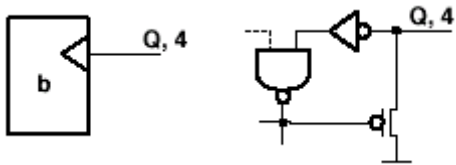
      |
N1 = ( p1 -> if full(2) then P2
      | g0 -> if full(2) then N0
      | else N2 fi
      | else G2 fi )

      |
P2 = ( s.p2 -> N2 )

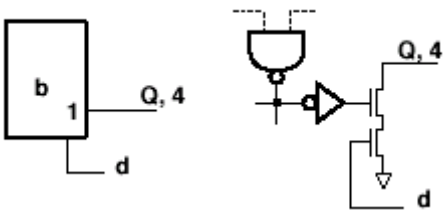
      |
end
  
```

FSM Snippet

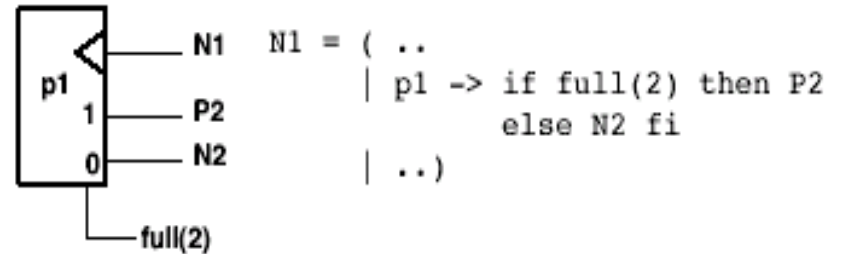
GasP Control Implementation



Input Part of GasP Cell

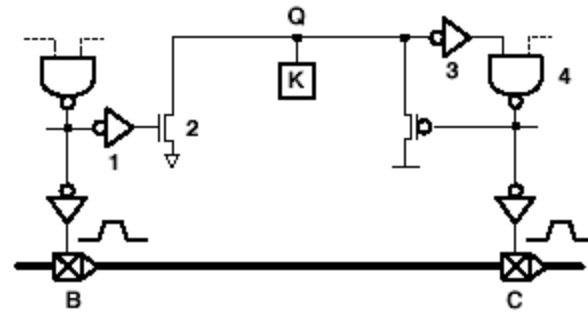
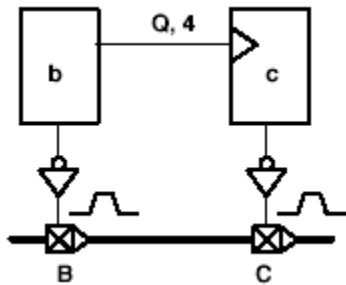


Output Part of GasP Cell



Sample GasP Cell for FSM
Conditional State Transition

GasP Interaction With Datapath

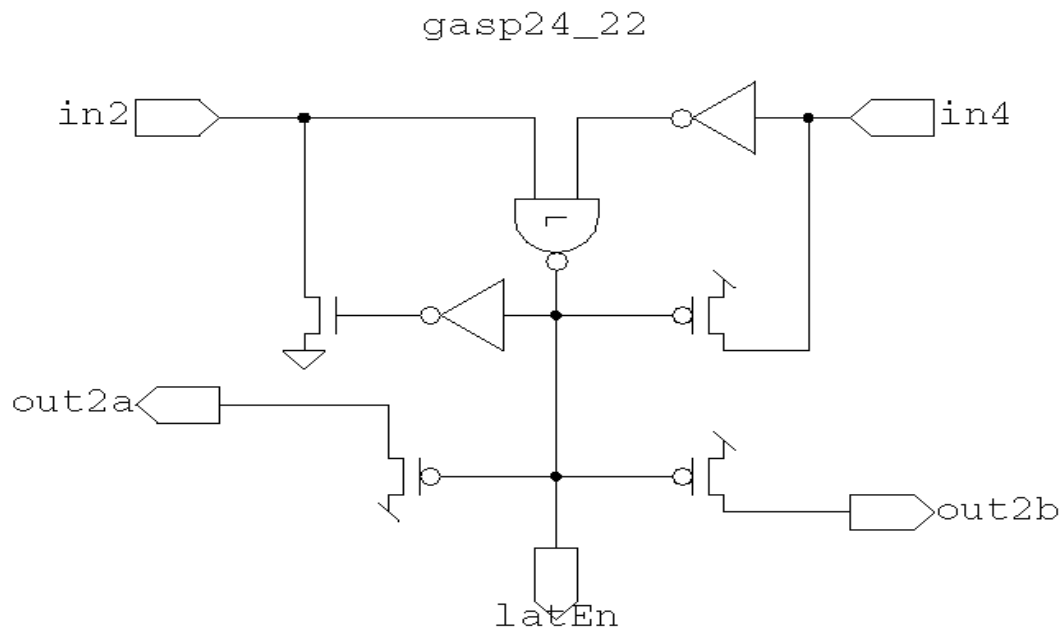


Successive GasP Modules
Controlling Latch Pulses

GasP Circuits -
Pulse Separation is
4 Gate Delays

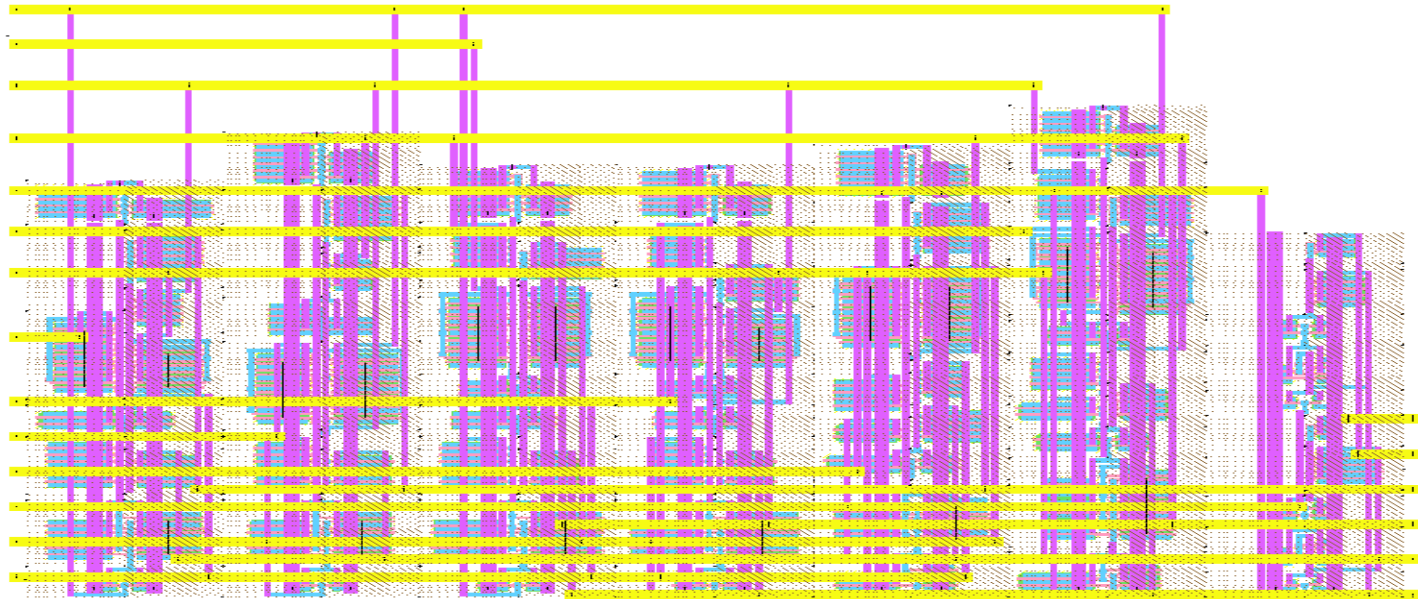
Automatic Transistor Sizing

- Based on the Logical Effort theory
- All gates are sized for equal delay
- Wire loads are back-annotated from layout
- Some iteration is necessary, but layout is also automated



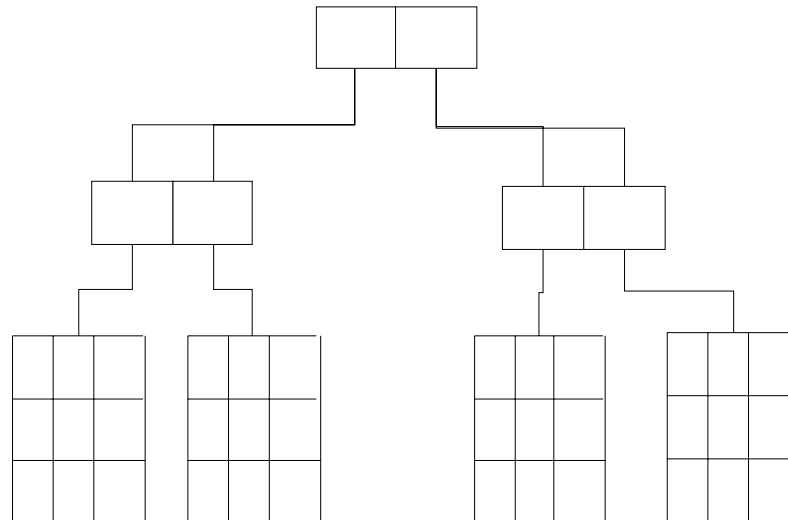
Automatic Layout

- CAD tool reads schematics and generates layout
- User provides hints to the program as to the general geometry
- Flexibility of quickly rescaling transistor sizes in the layout
- Area larger than hand layout, but design time greatly reduced



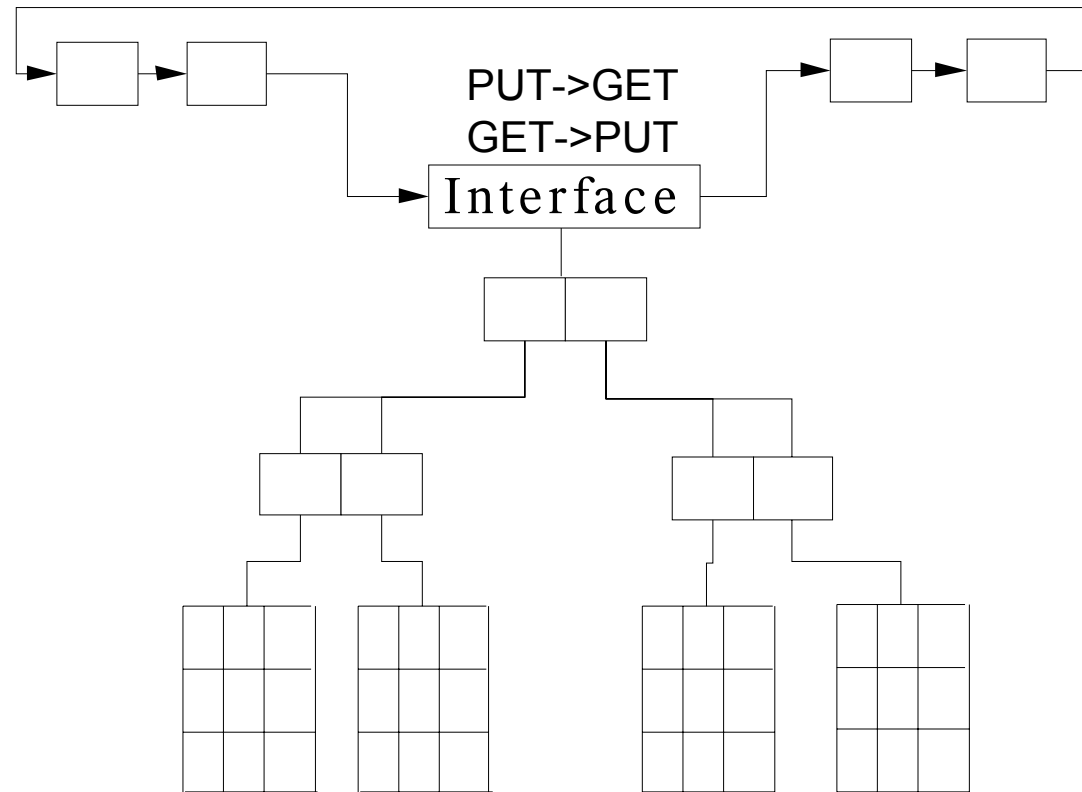
The Stack Chip

- 42-element stack
- 4-leaf tree stack with a linear stack at each leaf

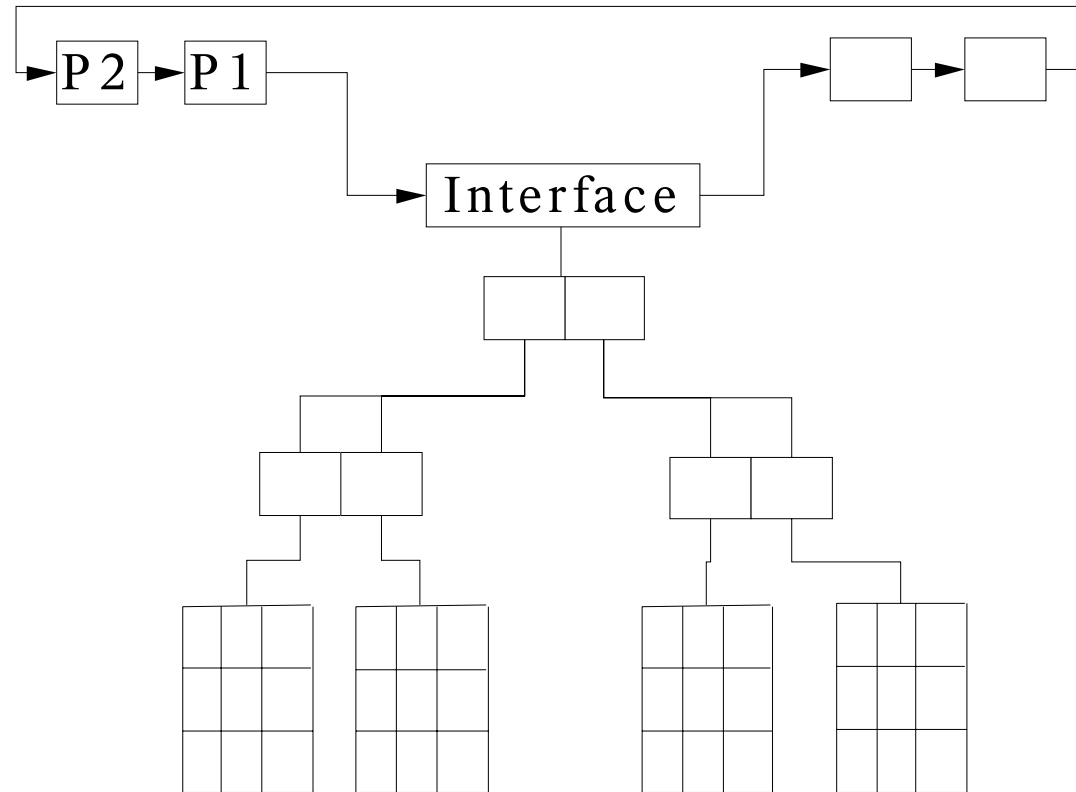


Stack Test Setup

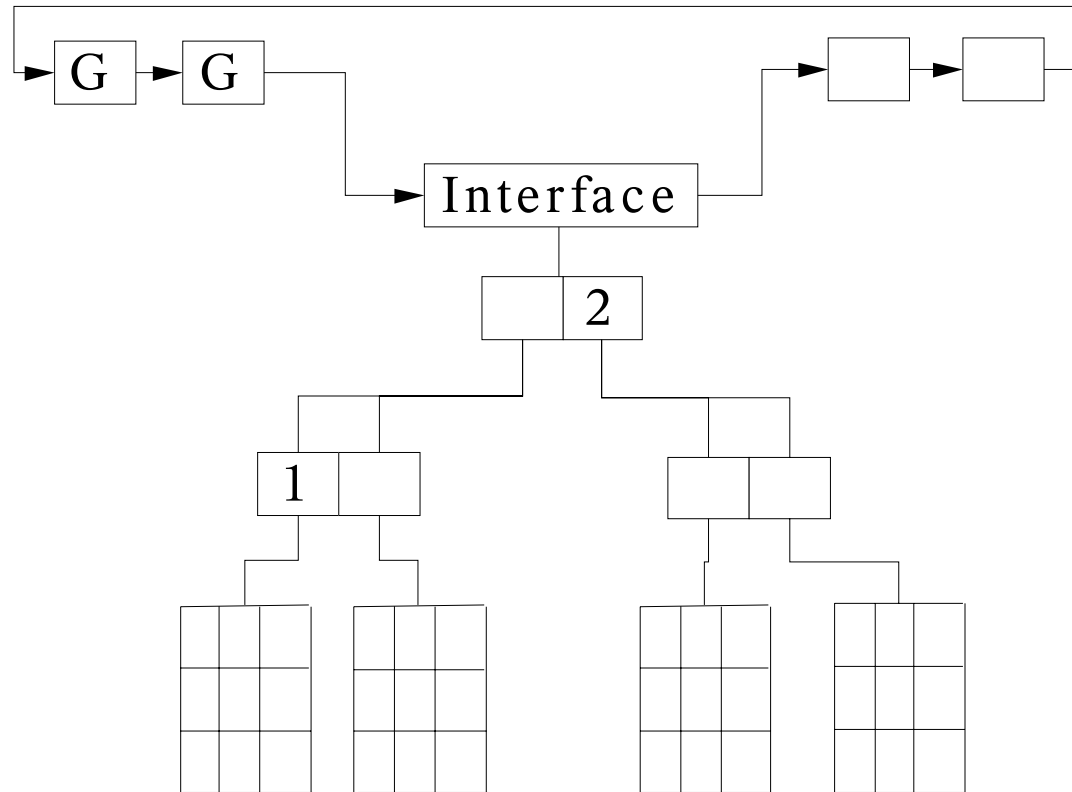
- 4-stage FIFO ring
- 42-element stack inserted between 2 FIFO stages



Ring Loaded with 2 Put Tokens

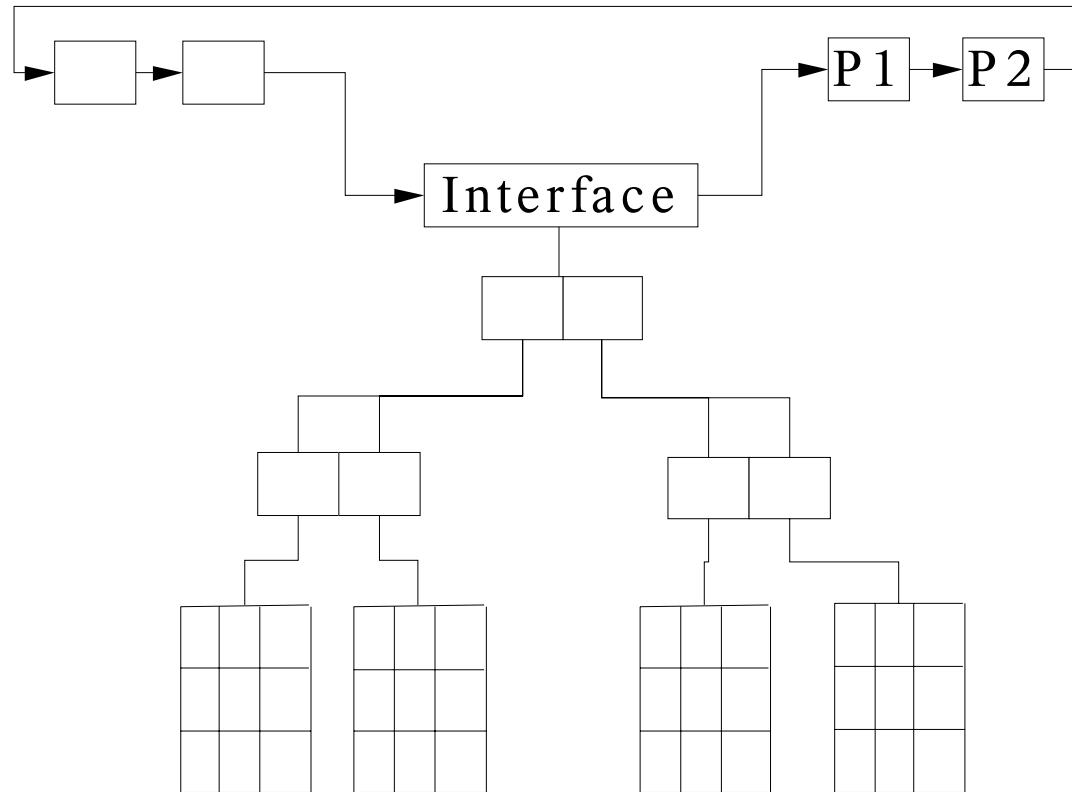


After 1st Pass

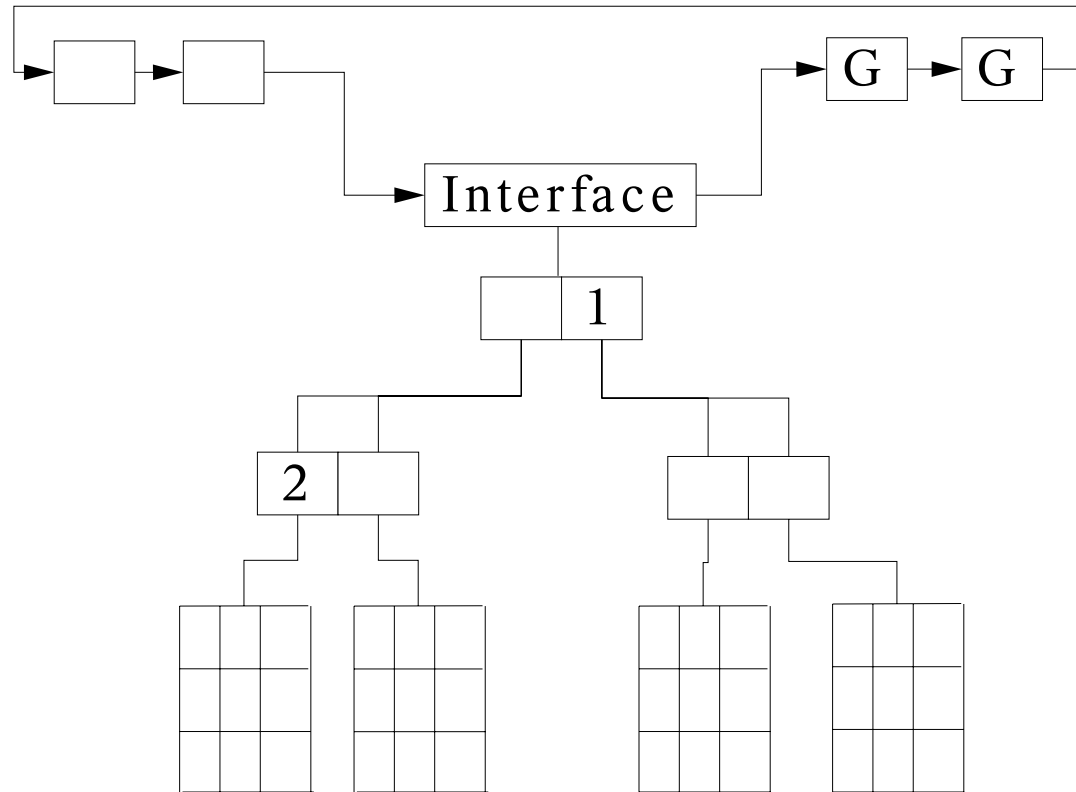


After 2nd Pass

- PUTs are now in reverse order



After 3rd Pass



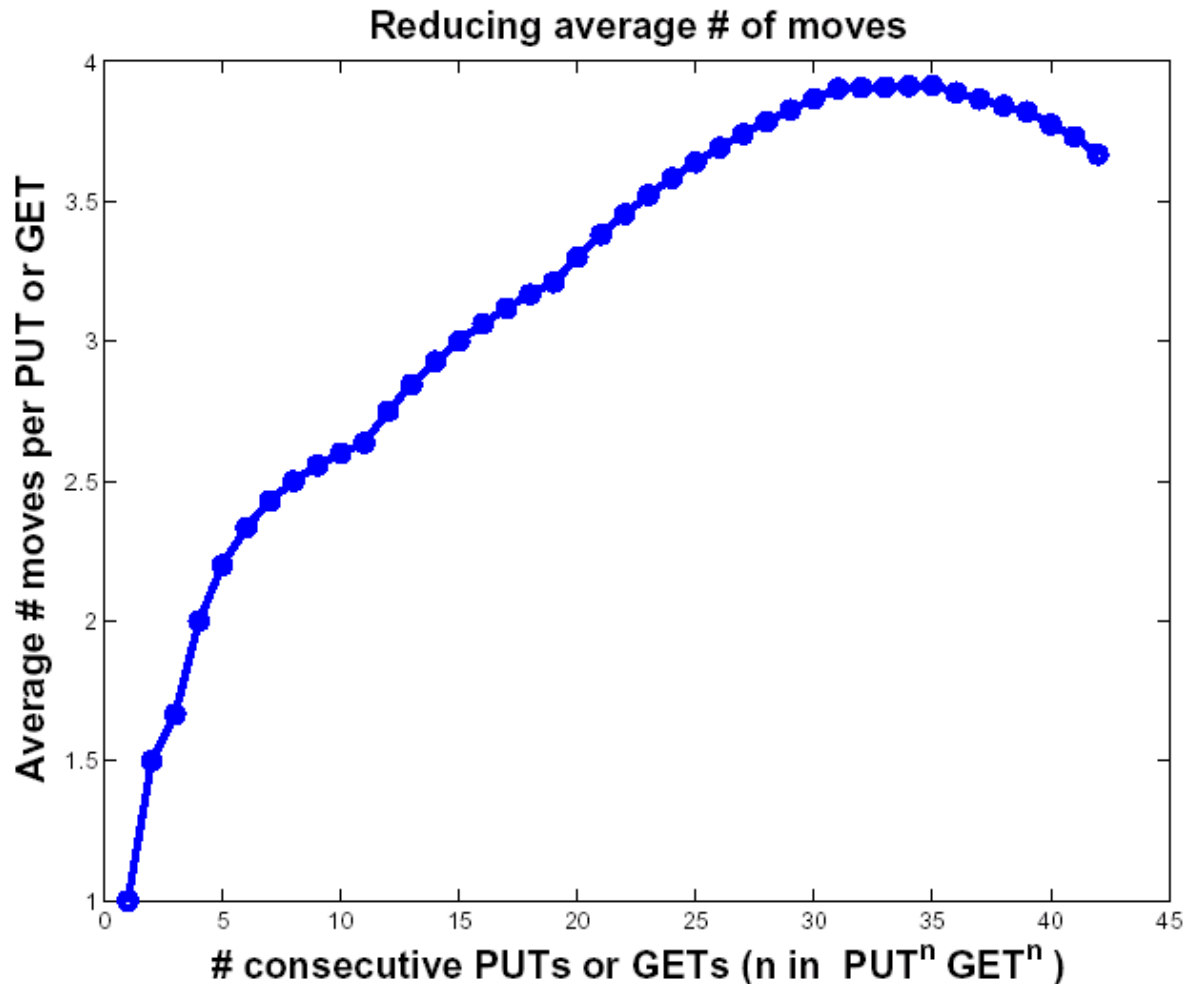
Stack Test Metrics



- **Functionality:** tokens in the ring can be read out and they should be the same as when loaded
- **Cycle Time:** counter attached to the ring increments whenever a token passes by
- **Power:** consumption should match mathematical prediction given for each test pattern

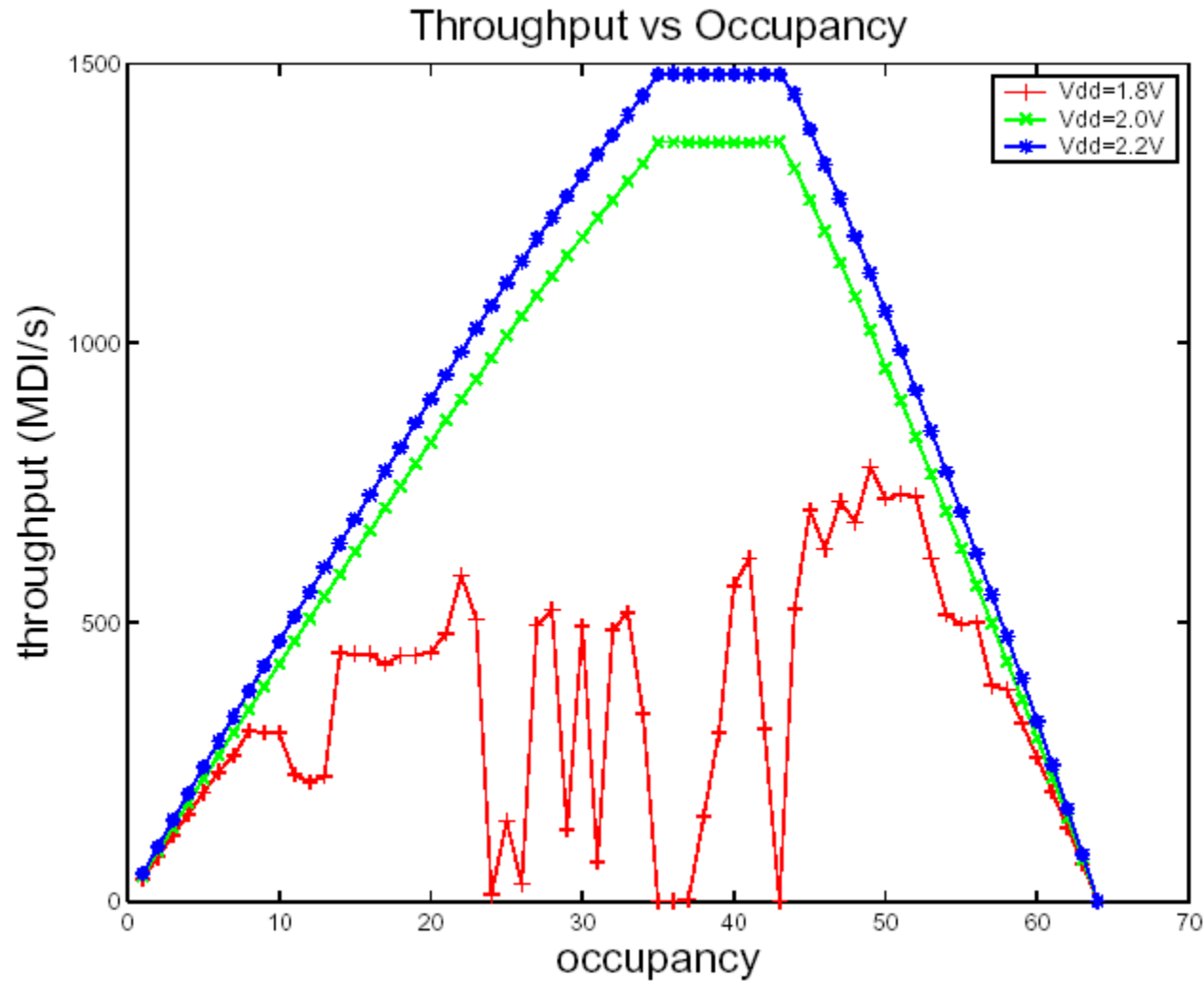
Reducing Data Movement

- On average, no more than 4 data moves per stack access
 - A simple linear stack averages 21 data moves per access



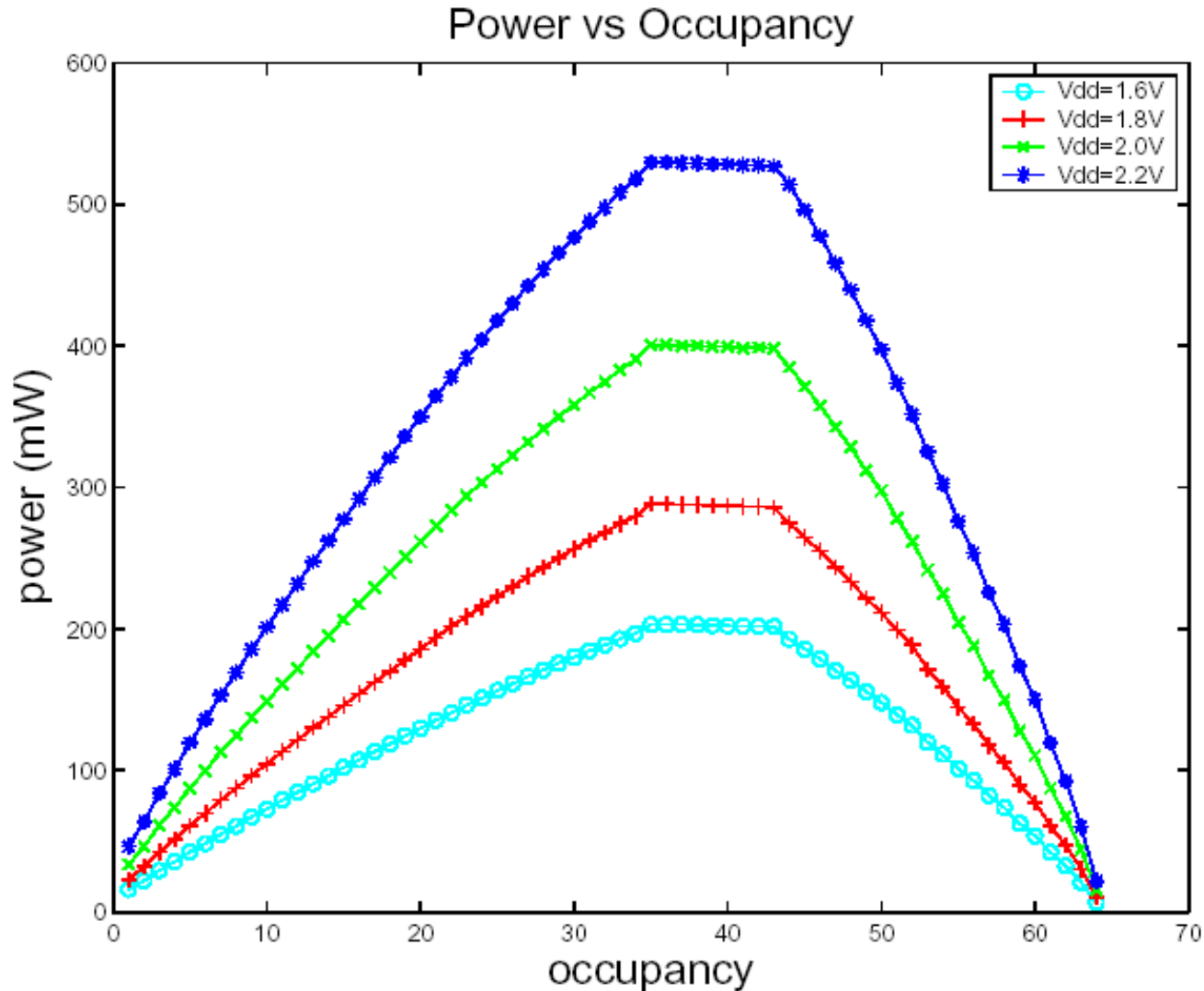
Chip Results – Ring Throughput

- Peak throughput of 1.25 GHz at 1.8V, matches simulation

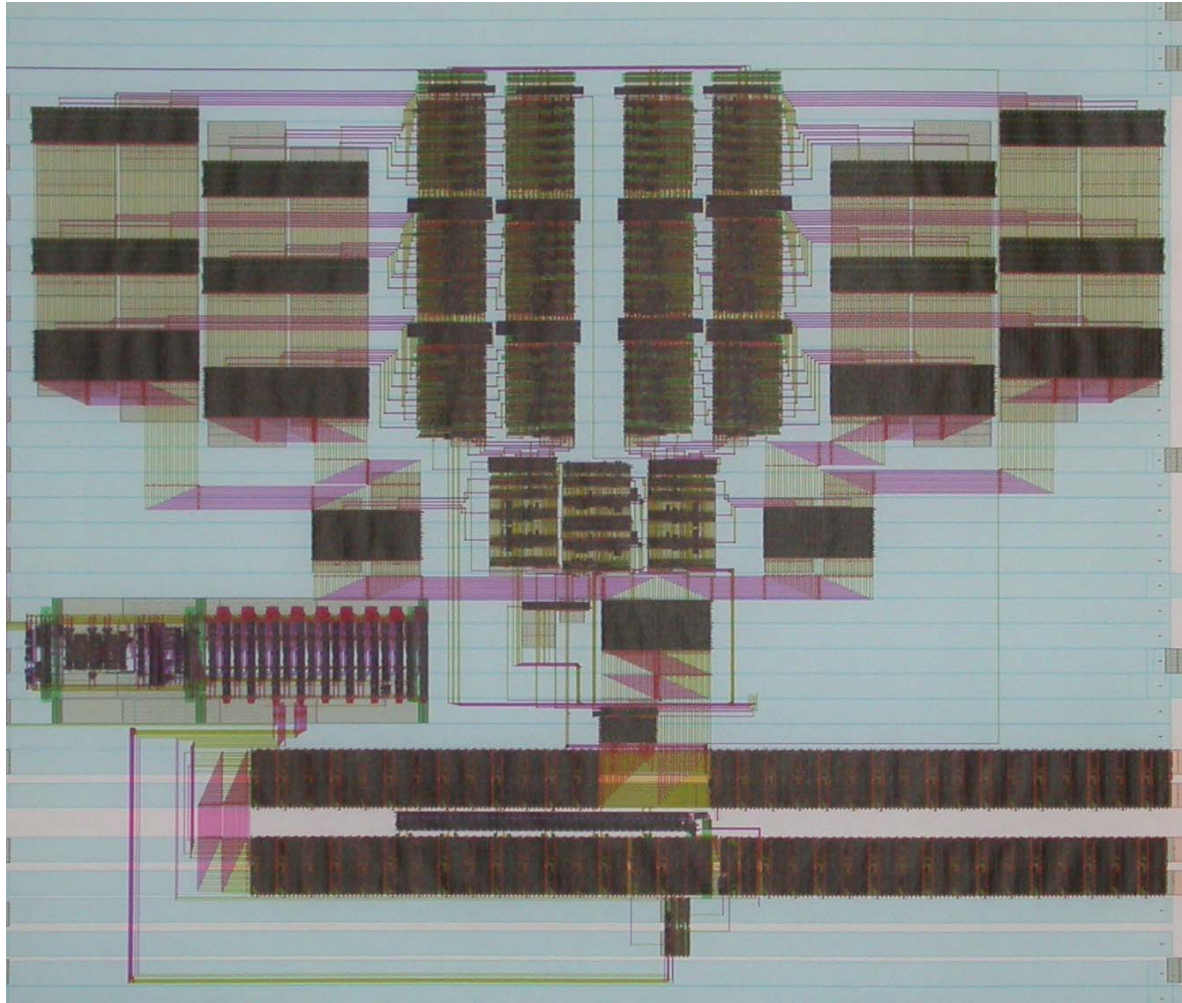


Chip Results – Total Power

- Energy used by tree control for moving 1 item: 16pJ



Test Chip Photo



Conclusions

- Asynchronous stack of arbitrary size and constant cycle time can be built
- Power savings due to significant reduction in data movements
- Demonstrated method of translating complex FSM to GasP implementation
- Automatic transistor sizing permits fine-grain control over cycle time and energy consumption
- Automated layout allows for quick development and iteration