

# System-on-a-chip and the Coming Design Revolution

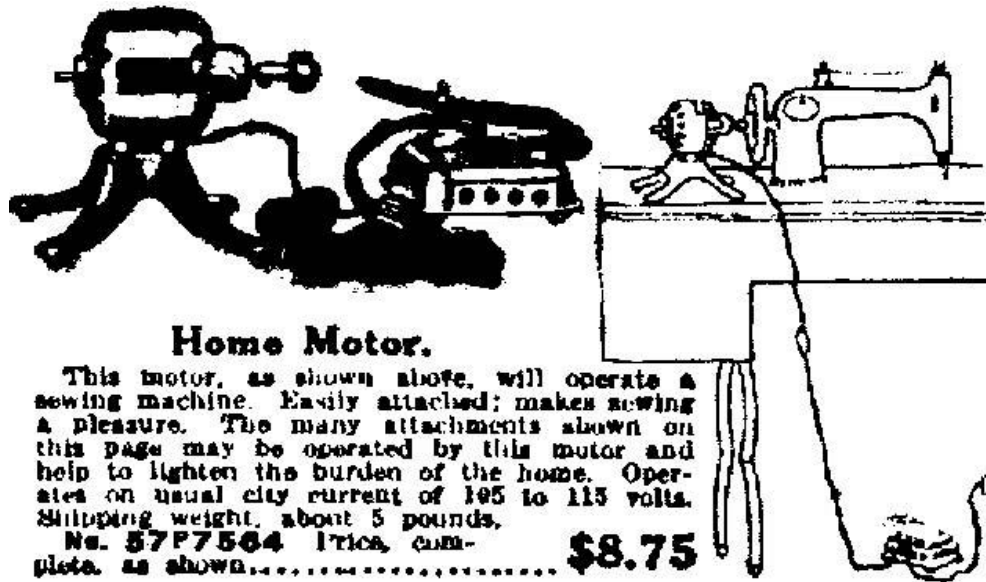
**Stephen A. Edwards**

Department of Computer Science,  
Columbia University

[www.cs.columbia.edu/~sedwards](http://www.cs.columbia.edu/~sedwards)

[sedwards@cs.columbia.edu](mailto:sedwards@cs.columbia.edu)

# 1918 Sears Roebuck Catalog



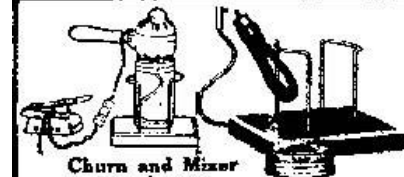
## Home Motor.

This motor, as shown above, will operate a sewing machine. Easily attached; makes sewing a pleasure. The many attachments shown on this page may be operated by this motor and help to lighten the burden of the home. Operates on usual city current of 105 to 115 volts. Shipping weight, about 5 pounds.  
**No. 57F7564 Price, complete, as shown..... \$8.75**



### Beater Attachment.

Whips cream and beats eggs, and many other uses will be found for these attachments when used in connection with the Home Motor. Parts include the stand, handle and the beater. Shipping weight, about 14 ounces.  
**No. 57F7585 Price..... \$1.30**



### Churn and Mixer Attachment.

Used in connection with the Home Motor, makes a small churn and mixer for which you will find many uses. The attachments include the base, supports, mixer, handle and special cover for jar. Shipping weight, about 1 1/2 pounds.  
**No. 57F7582 Price..... \$1.30**



### Fan Attachment.

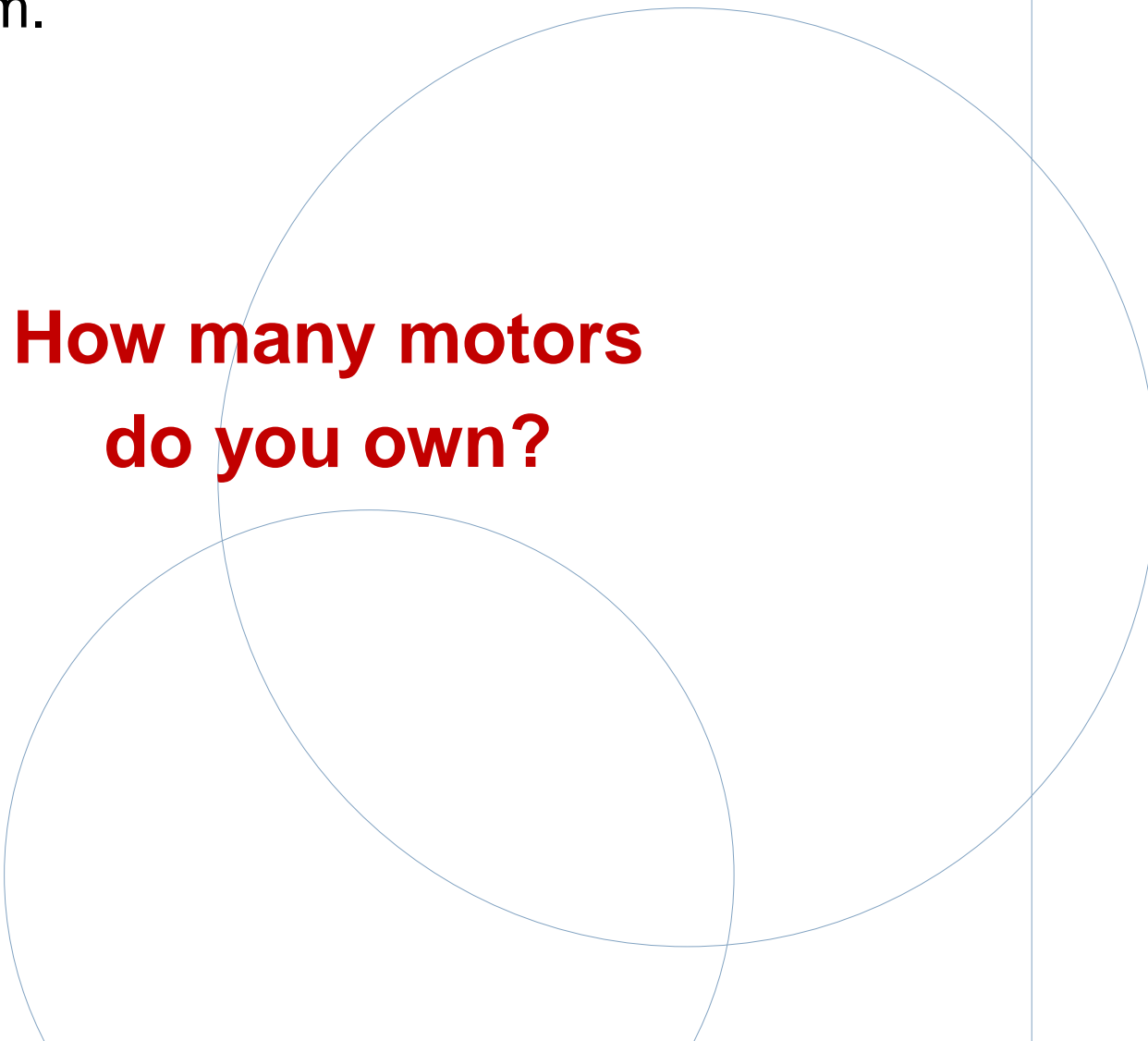
Includes fan and guard which can be quickly attached to Home Motor, and will be a great comfort in hot weather. Shipping weight, about 14 ounces.  
**No. 57F6215 Price..... \$1.30**

About \$100 in today's dollars.

From Donald Norman, *The Invisible Computer*, 1998.

# What happened to Home Motors?

Motors became cheap enough to embed in any appliance that needed them.



**How many motors  
do you own?**

# 2000 MacMall Catalog

## The New iMac! 350MHz G3 Processor and Stereo Sound!

Check out the affordable new iMac with its faster processor and a bigger hard disk. You can quickly surf the Internet, send and receive e-mail and play CDs.

### Features and Benefits:

- 350MHz G3 PowerPC processor
- 64MB SDRAM; support up to 1GB
- 7GB Ultra ATA hard disk drive
- Slot-loading 24X speed CD-ROM drive
- Built-in 15" (13.8" viewable)
- Resolutions up to 1024 x 768
- Built-in 56K V.90 modem
- 2 USB ports
- 10/100Base-T Ethernet

Own this iMac for as low as \$23/month with the NEW MacMall EZ Payment Plan!



350MHz iMac™  
only \$799 #953100 (Indigo only)  
With FREE 64MB\*

See page 13 for details on the new MacMall EZ Payment Plan.  
\*400 MB 7000 hard drive. 70 GB-10GB 50K modem.

## Great Add-ons!

LACIE

LaCie 8x4x32  
External FireWire  
CD-RW Drive



only \$349.99 #56665  
See page 45.

#52201 Toast 4 Deluxe

100MB USB  
Zip Drive

only \$69.99

#88739  
See page 38.  
100% non-recycled.



OLYMPUS

D-490  
Zoom  
Camera



only \$499.99 #951256  
See page 33.

How many computers do you own?

# What will the SoCs of the future be?

Hint:



# Hidden Computers



Casio  
Camera  
Watch



Nokia 7110  
Browser  
Phone



Sony  
Playstation 2



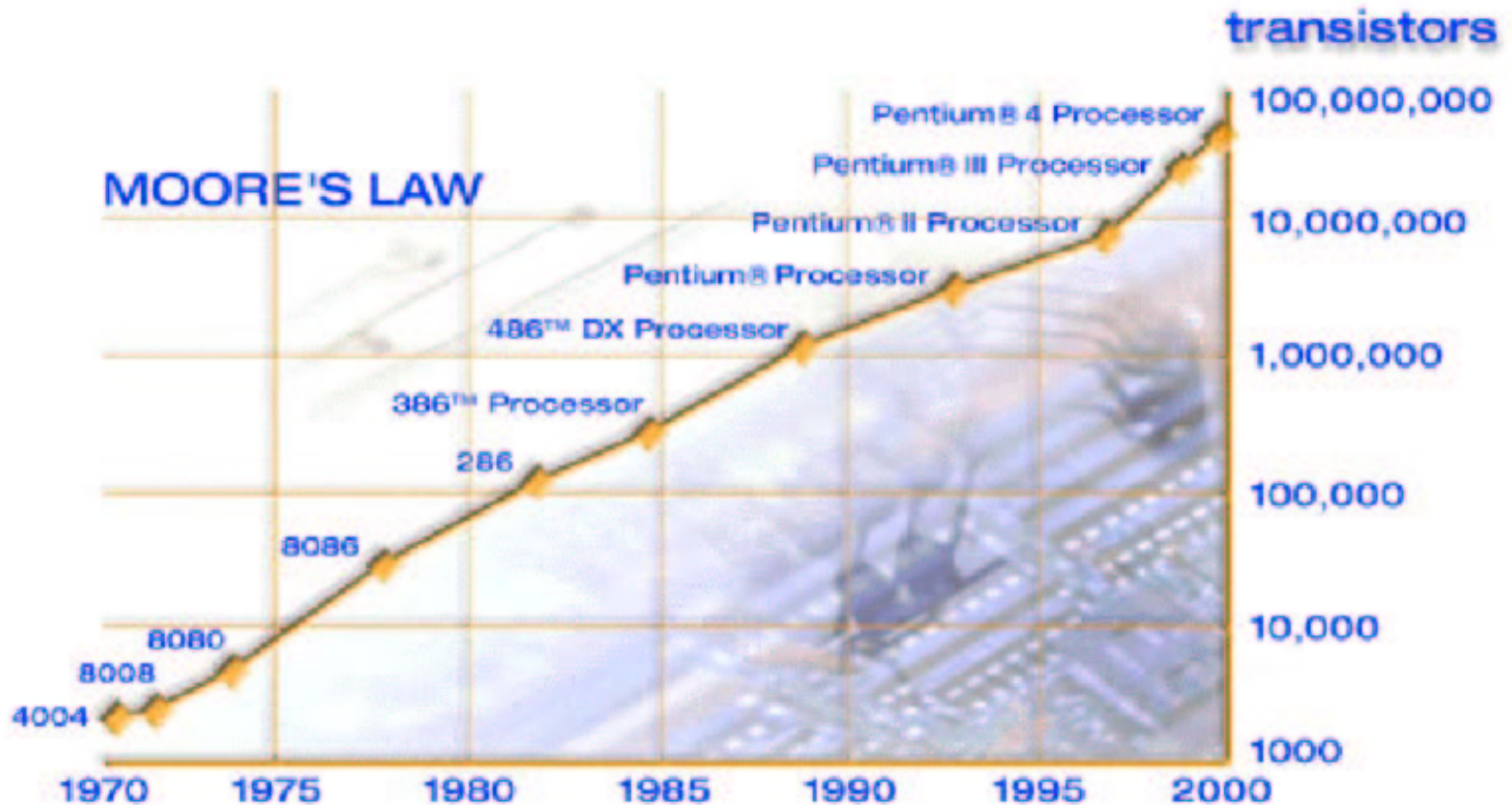
Philips  
DVD Player



Philips  
TiVo Recorder



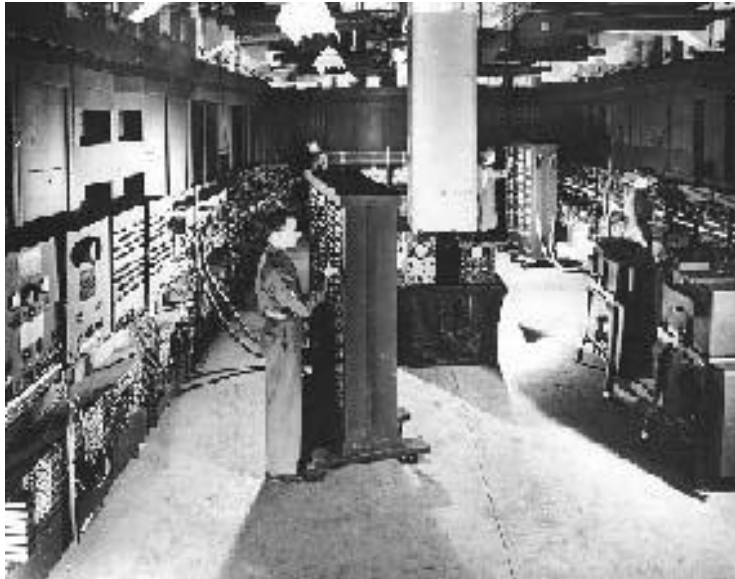
# Transistor Cost Continues Plummeting



Each Pentium sold for about \$600 initially.

Source: Intel

# Computers' Changing Role



Environment and humans  
subservient to computer

Simple peripherals



Computers subservient to  
humans and the  
environment

Complex peripherals



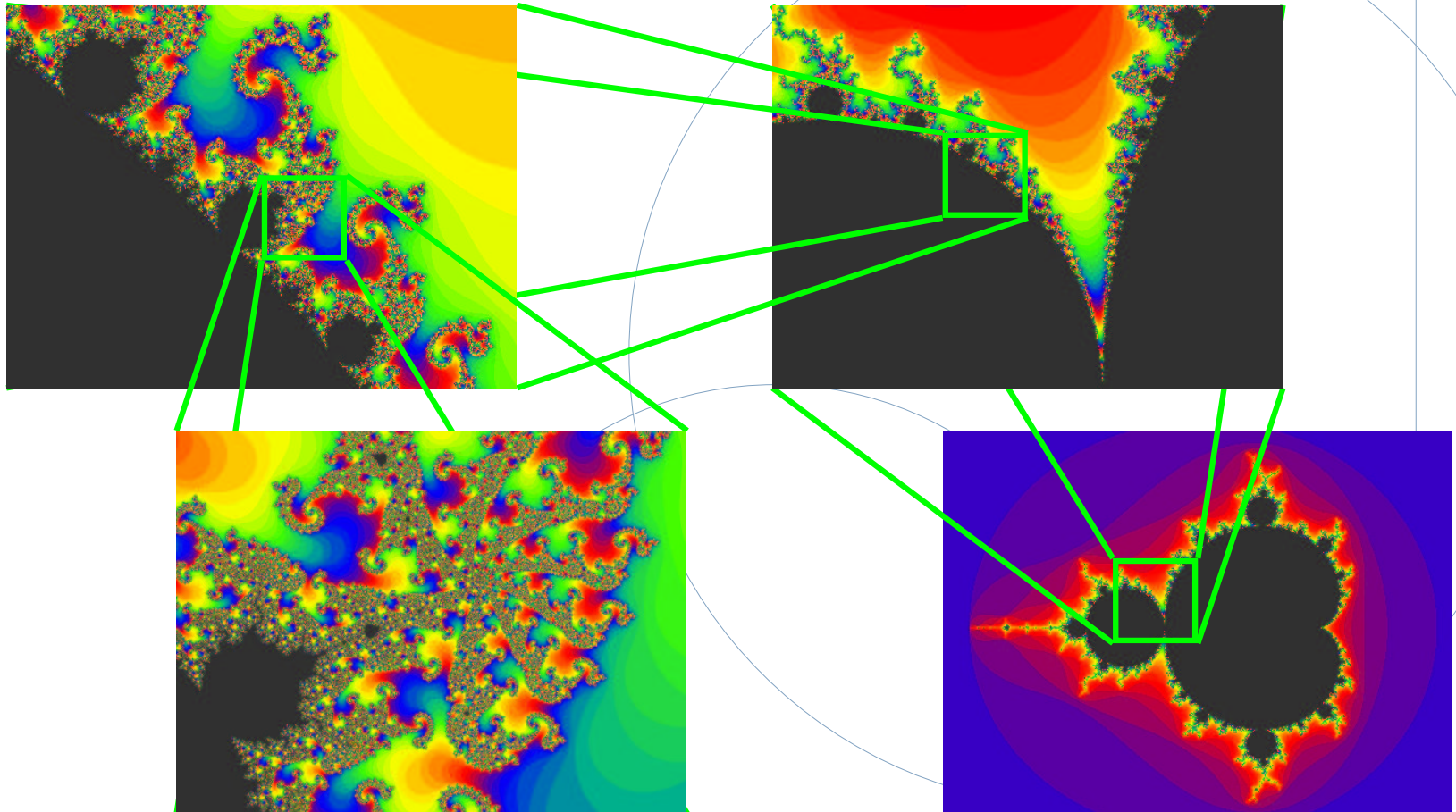
# Embedded System Challenges

## Real-time Deadlines



# Embedded System Challenges

## Complexity



# Software complexity growing

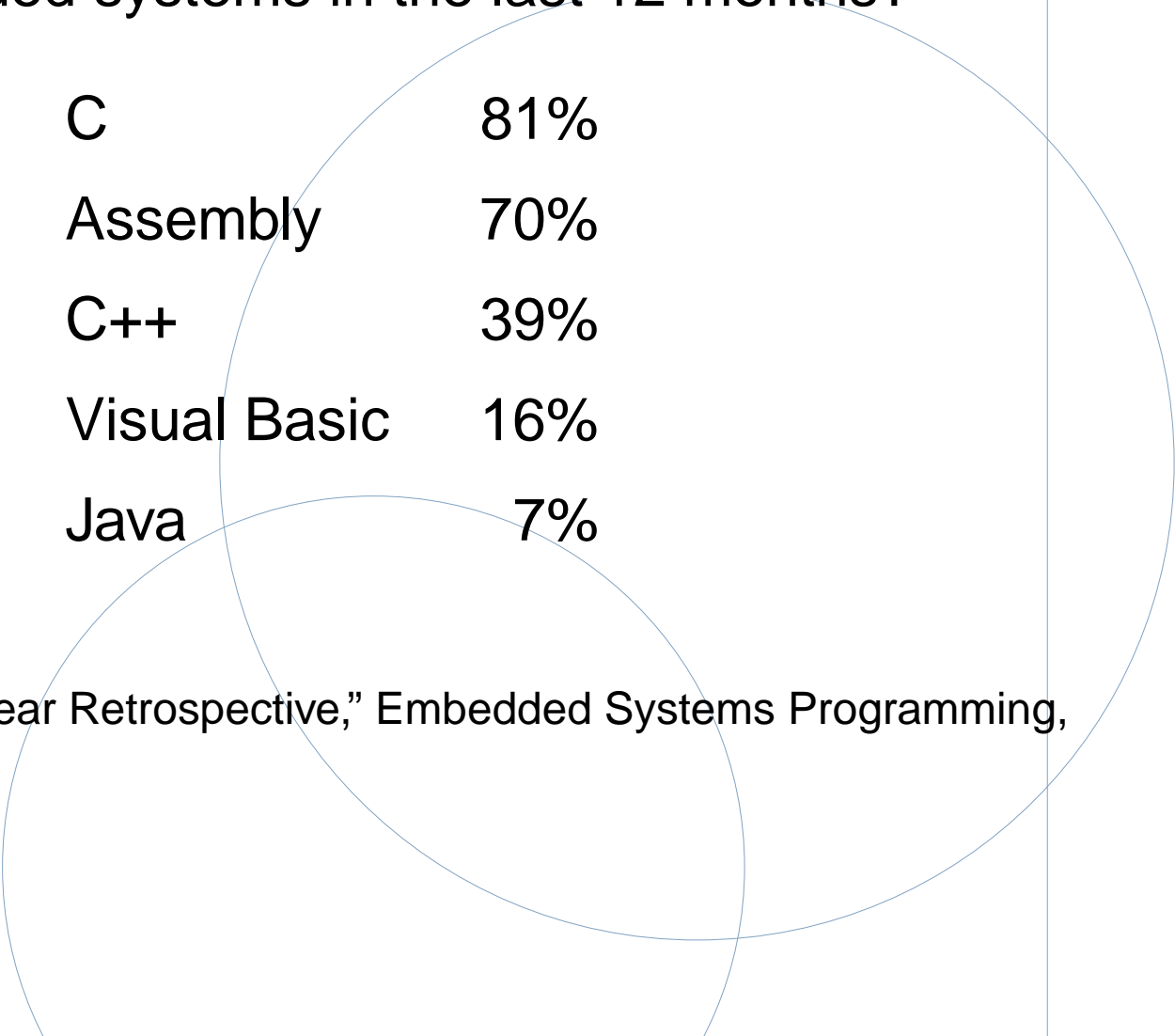
## Size of Typical Embedded System

1985	13 kLOC	
1989	21 kLOC	↓ 44 % per year
1998	1 MLOC	
2000	2 MLOC	
2008	16 MLOC	≈ Windows NT 4.0
2010	32 MLOC	≈ Windows 2000

Source: "ESP: A 10-Year Retrospective," Embedded Systems Programming, November 1998

# Written in stone-age languages

“Which of the following programming languages have you used for embedded systems in the last 12 months?”



C	81%
Assembly	70%
C++	39%
Visual Basic	16%
Java	7%

Source: “ESP: A 10-Year Retrospective,” Embedded Systems Programming, November 1998

# Embedded System Challenges

## Concurrency



Photo by Thomas Danoghue

# Existing Techniques



...aren't up to the task.

- Existing multi-threaded concurrency models  
...are completely unstructured  
The “goto” of control
- Most real-time scheduling  
...ignores communication aspects

We need some alternatives!



# An Alternative: Esterel

Domain-specific language for safety-critical, real-time systems.

Uses a synchronous model of time that is deterministic and provides precise control over time.

Timing verification becomes checking a single worst-case-execution-time bound.

# Timing

## Java

```
class PClock
  implements Runnable {
  public void run() {
    for (;;) {
      java.util.Date now =
        new java.util.Date();
      System.out.
        println(now.toString());
      try {
        Thread.currentThread().
          sleep(1000);
      } catch (InterruptedException e) {}
    }
  }
}

public class Clock {
  public static void
  main(String args[]) {
    Thread t =
      new Thread(new PClock());
    t.start();
  }
}

$ java Clock
Sat Sep 14 13:04:27 EDT 2002
Sat Sep 14 13:04:29 EDT 2002
Sat Sep 14 13:04:30 EDT 2002
Sat Sep 14 13:04:31 EDT 2002
```

## Esterel

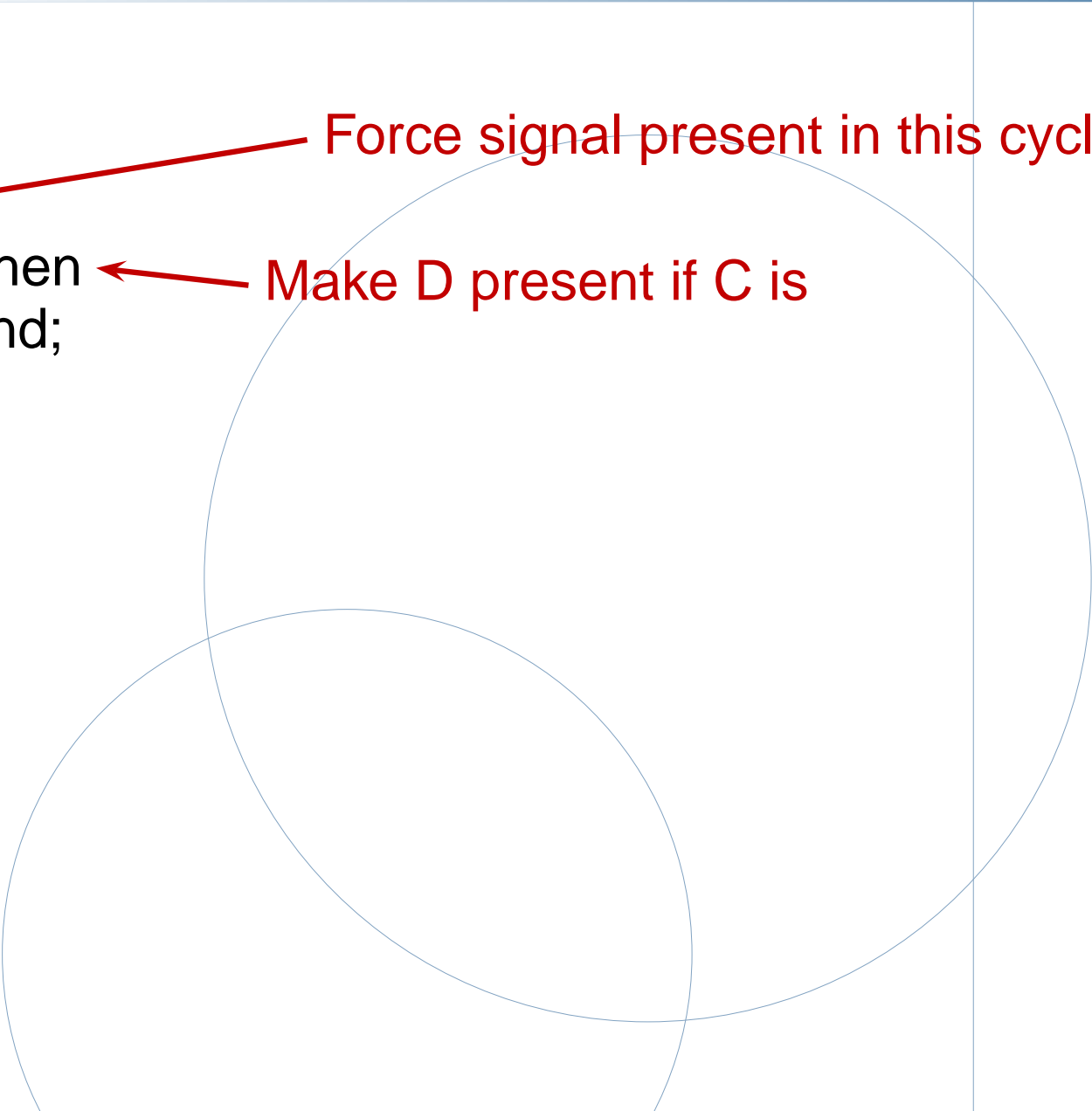
```
every 1000 MS do
  emit SECOND
end
```

Just works

A Leap Second?

# An Example

emit B; ← Force signal present in this cycle  
present C then ← Make D present if C is  
emit D end;

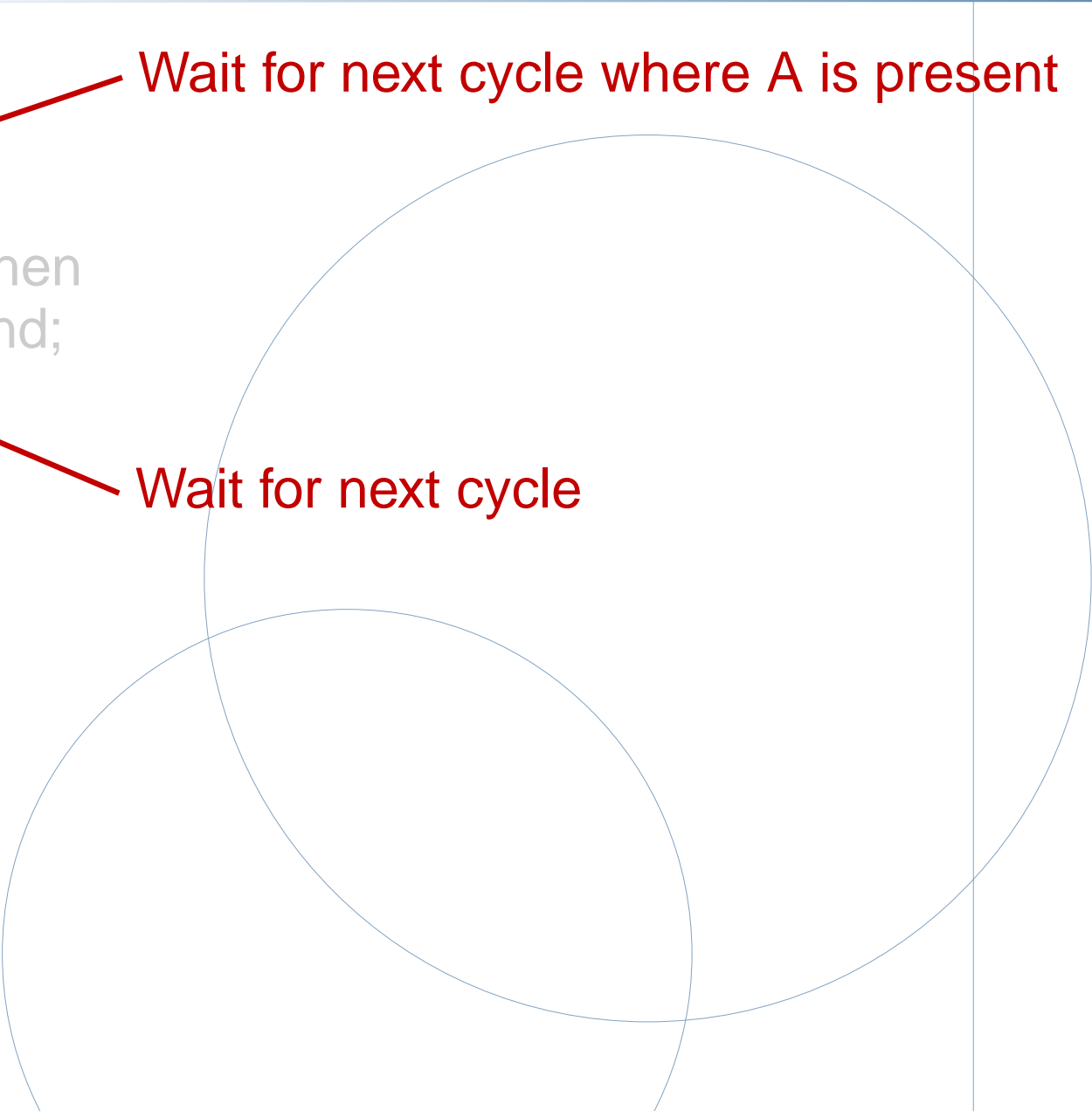
A diagram consisting of two overlapping circles and a vertical line. The circles are light blue and overlap in the lower-left quadrant. The vertical line is a thin blue line on the right side of the image, extending from the top to the bottom.

# An Example

```
await A;
emit B;
present C then
  emit D end;
pause
```

Wait for next cycle where A is present

Wait for next cycle

The diagram consists of two overlapping circles and a vertical line. The circles are light blue and overlap in the center. The vertical line is also light blue and passes through the right side of the circles. The text and arrows are positioned to the left of the circles.

# An Example

```
loop ← Infinite Loop
  await A;
  emit B;
  present C then
    emit D end;
  pause
end
```



# An Example

```
loop  
  await A;  
  emit B;  
  present C then  
    emit D end;  
  pause  
end
```



**Run Concurrently**

```
loop  
  present B then  
    emit C end;  
  pause  
end
```

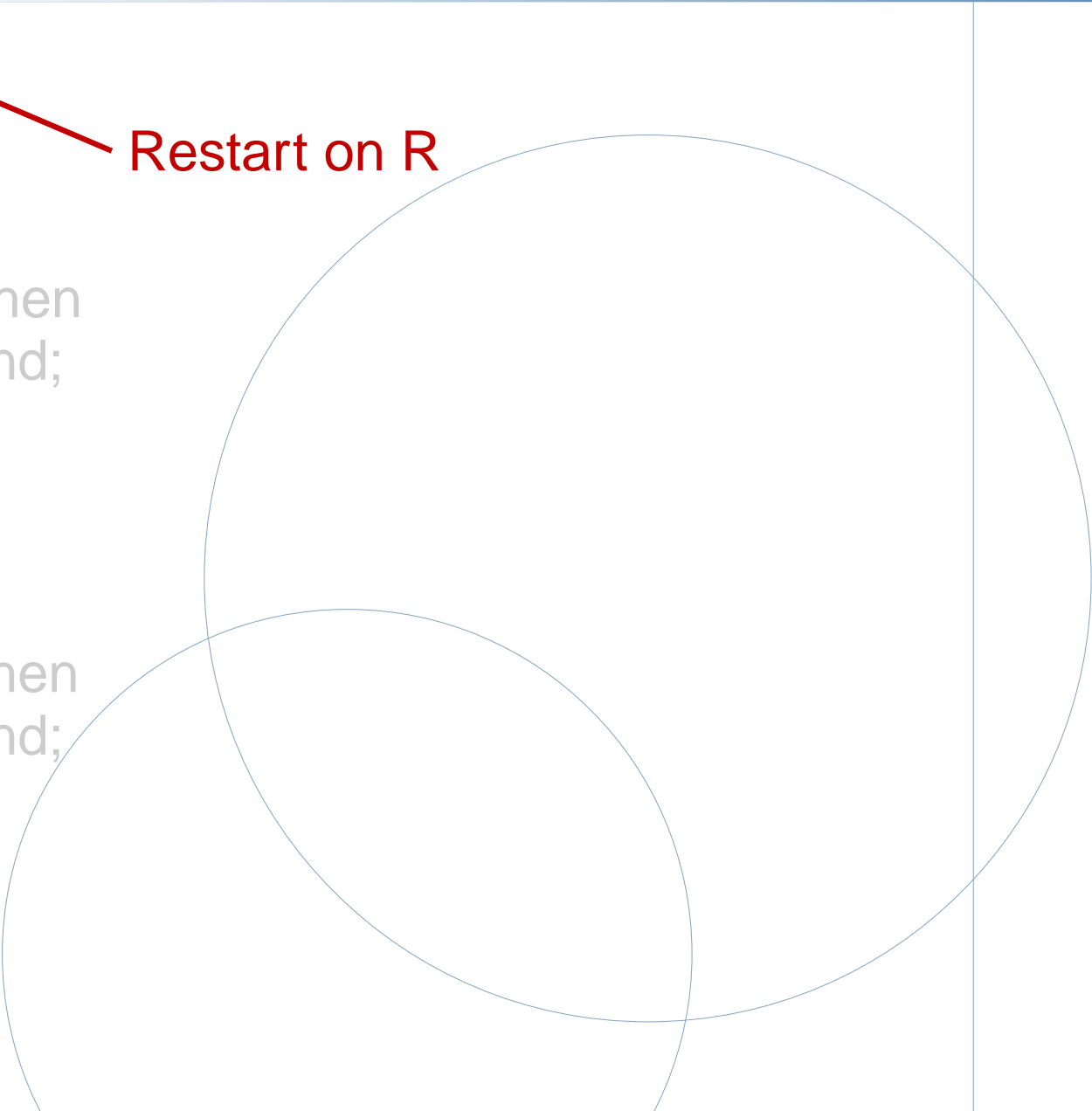




# An Example

```
every R do  
  loop  
    await A;  
    emit B;  
    present C then  
      emit D end;  
    pause  
  end  
||  
  loop  
    present B then  
      emit C end;  
    pause  
  end  
end
```

Restart on R

A diagram consisting of two overlapping circles and a vertical line. The circles are light blue and overlap in the center. The vertical line is also light blue and runs from the top to the bottom of the frame, positioned to the right of the circles.

# An Example

```
every R do
  loop
    await A;
    emit B;
    present C then
      emit D end
    pause
  end
||
loop
  present B then
    emit C end;
  pause
end
end
```



Same-cycle bidirectional communication

# An Example

```
every R do
  loop
    await A;
    emit B;
    present C then
      emit D end;
    pause
  end
||
  loop
    present B then
      emit C end;
    pause
  end
end
```

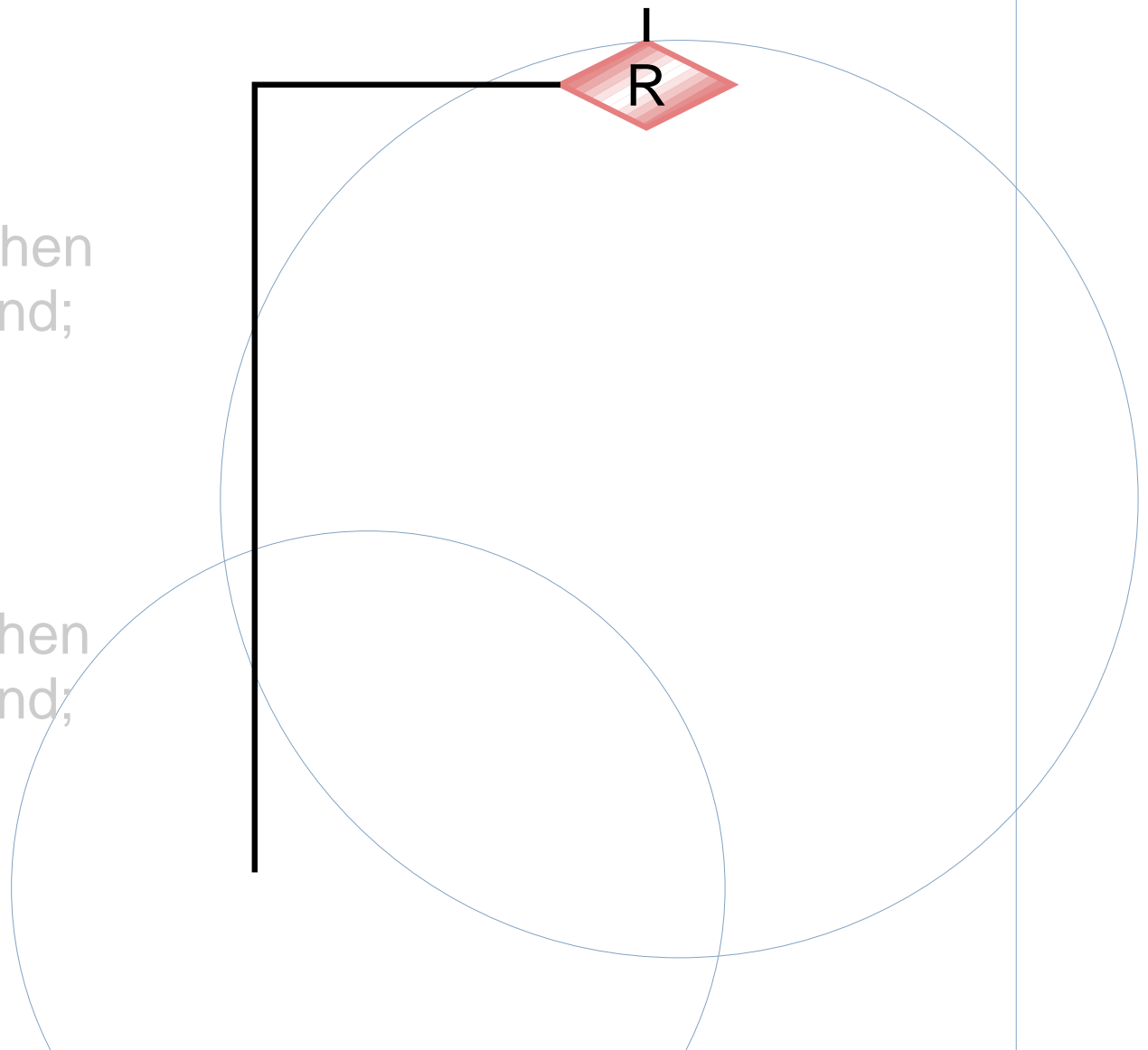
Good for hierarchical FSMs

Bad at manipulating data

Hardware Esterel variant  
proposed to address this

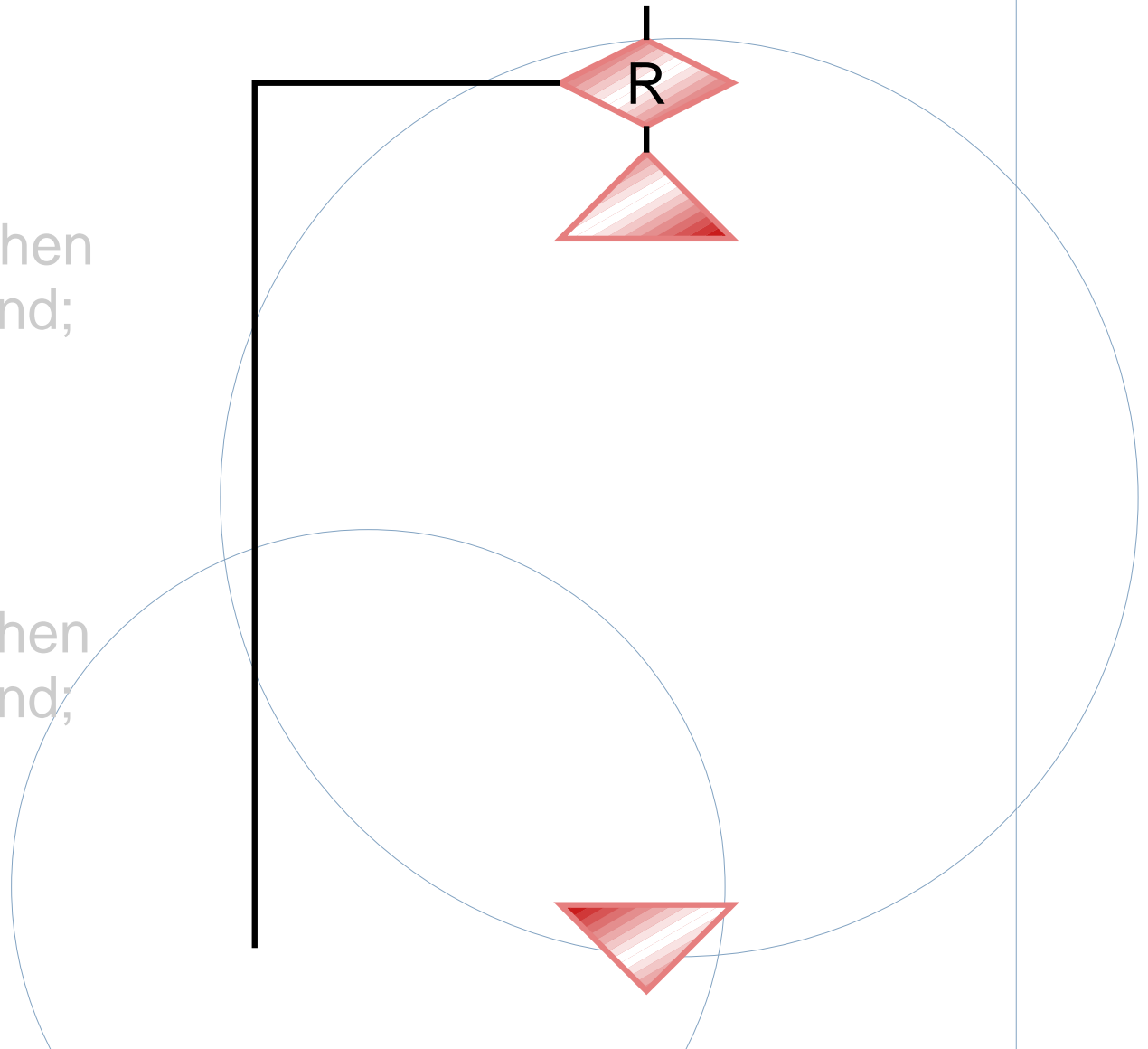
# Translate every

```
every R do
  loop
    await A;
    emit B;
    present C then
      emit D end;
    pause
  end
||
  loop
    present B then
      emit C end;
    pause
  end
end
```



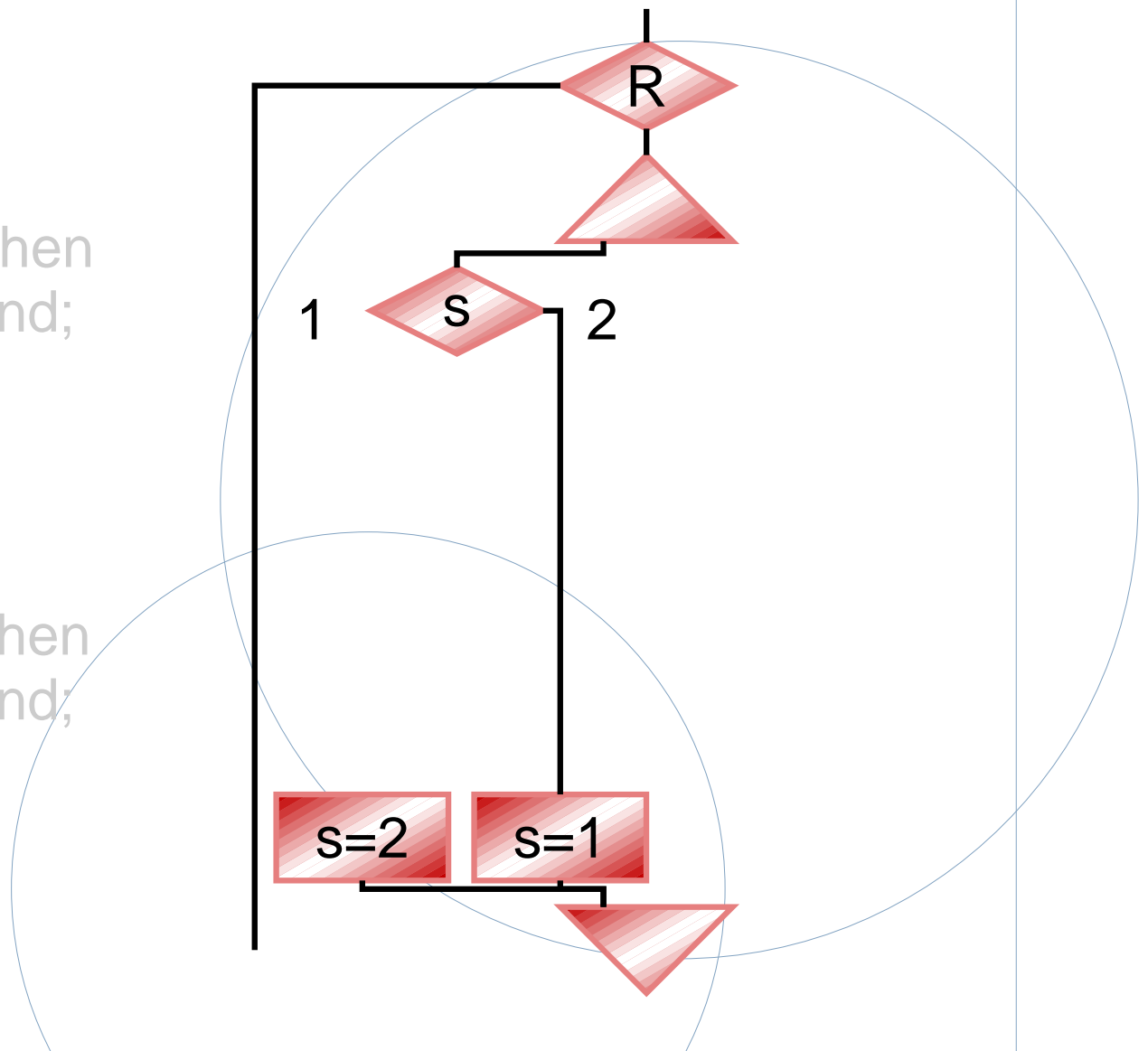
# Add Threads

```
every R do
  loop
    await A;
    emit B;
    present C then
      emit D end;
    pause
  end
||
  loop
    present B then
      emit C end;
    pause
  end
end
```



# Split at Pauses

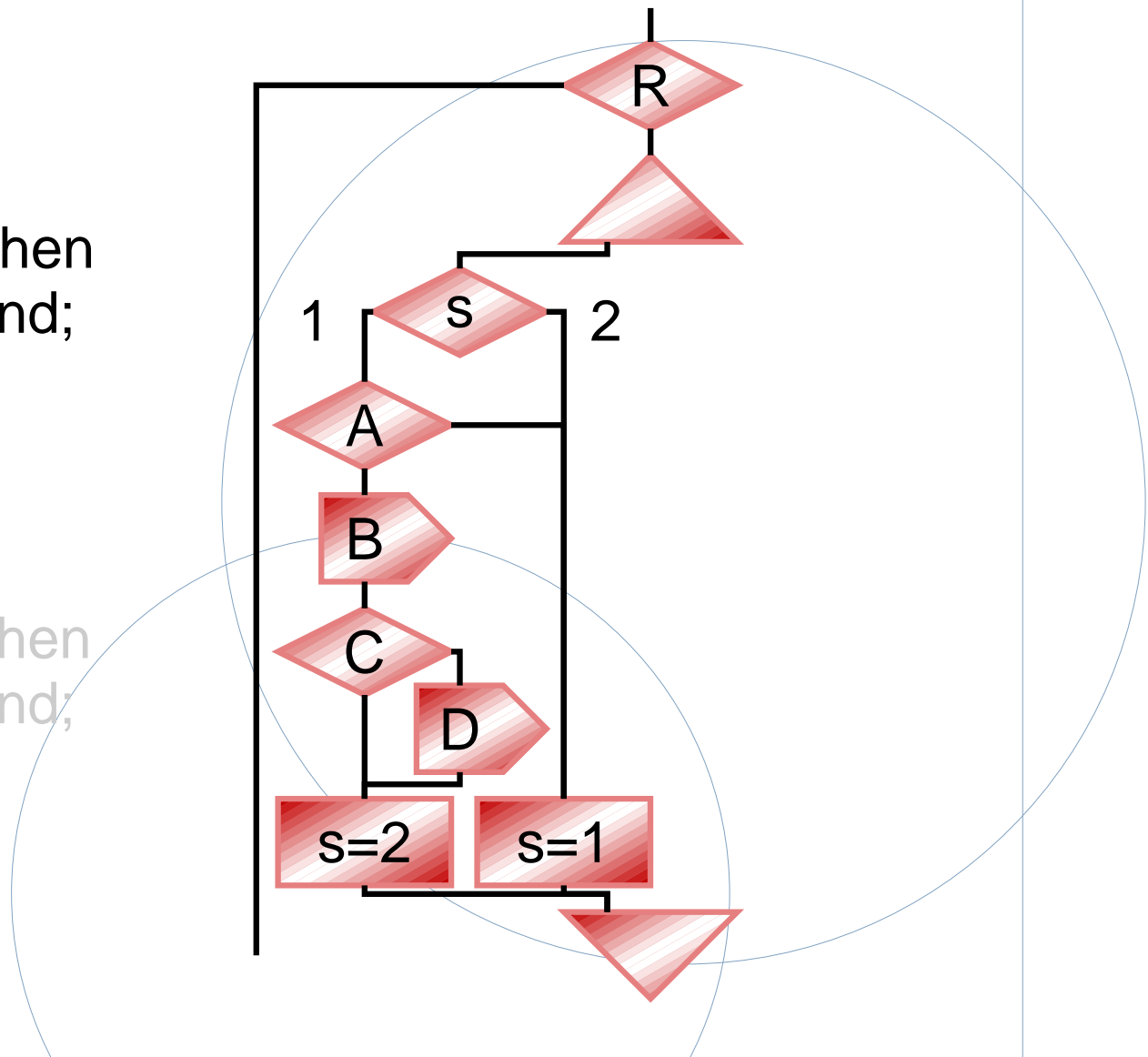
```
every R do
  loop
    await A;
    emit B;
    present C then
      emit D end;
    pause
  end
||
loop
  present B then
    emit C end;
  pause
end
end
```





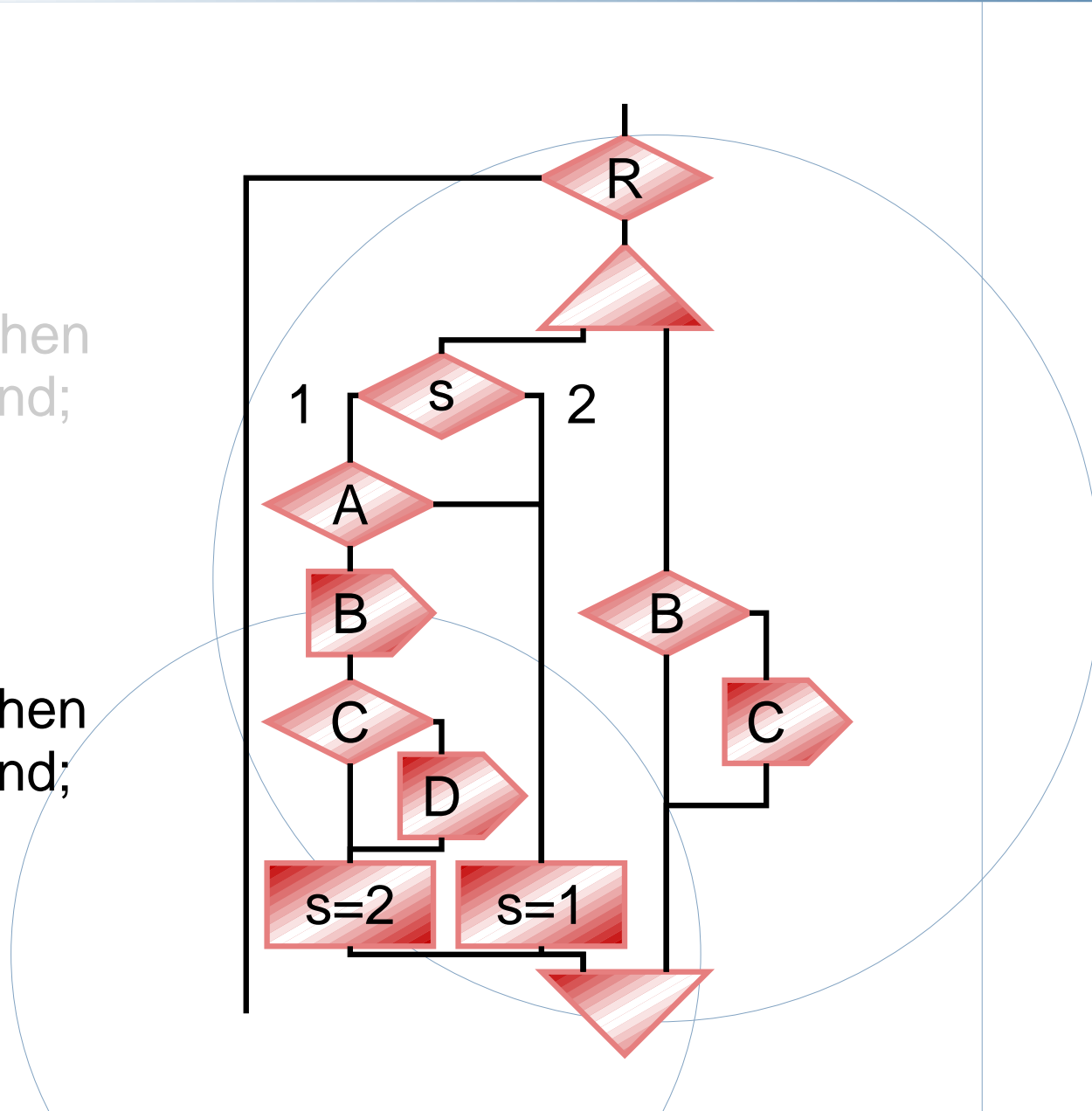
# Add Code Between Pauses

```
every R do
  loop
    await A;
    emit B;
    present C then
      emit D end;
    pause
  end
||
  loop
    present B then
      emit C end;
    pause
  end
end
```



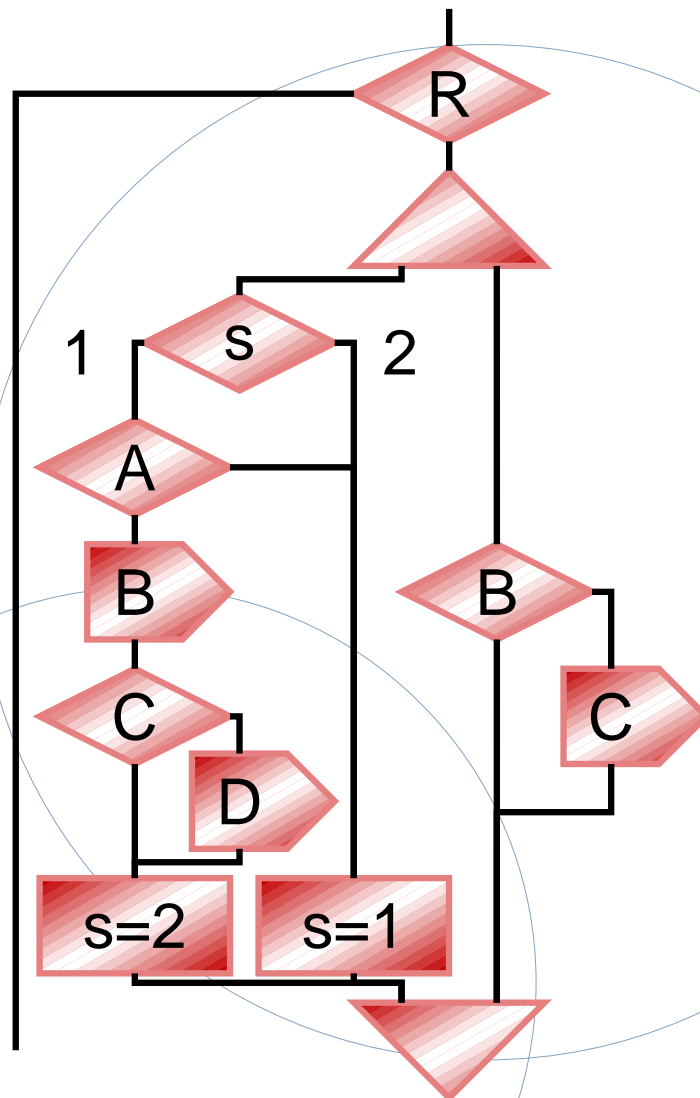
# Translate Second Thread

```
every R do
  loop
    await A;
    emit B;
    present C then
      emit D end;
    pause
  end
||
loop
  present B then
    emit C end;
  pause
end
end
```



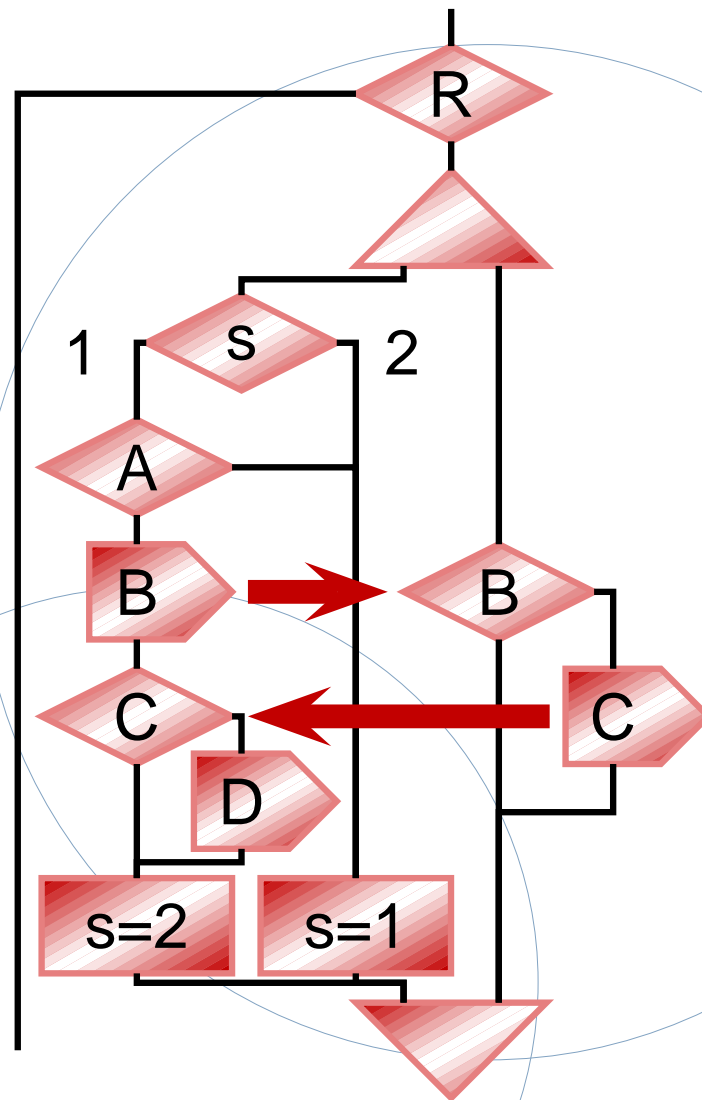
# Finished Translating

```
every R do
  loop
    await A;
    emit B;
    present C then
      emit D end;
    pause
  end
||
loop
  present B then
    emit C end;
  pause
end
end
```

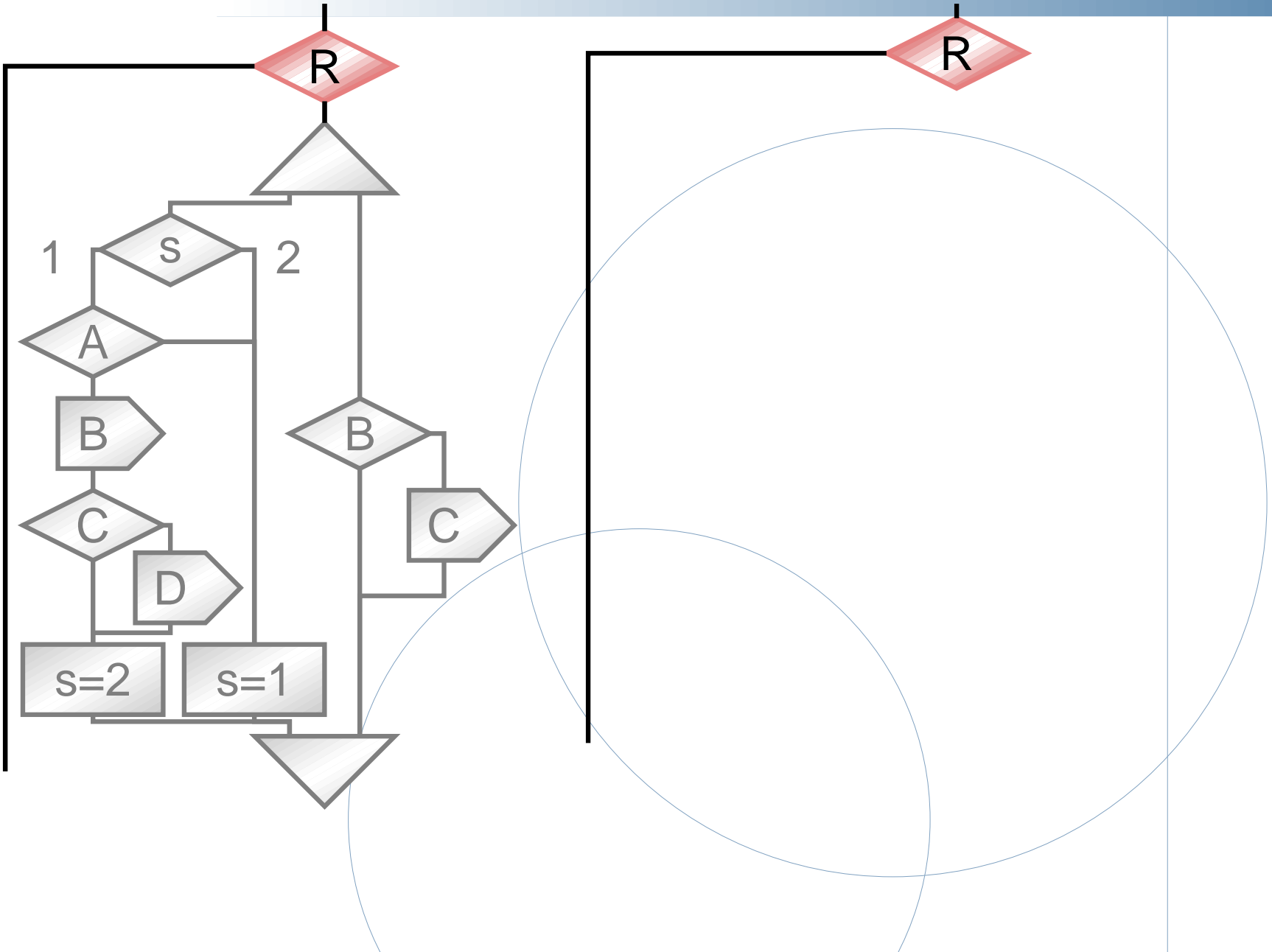


# Add Dependencies and Schedule

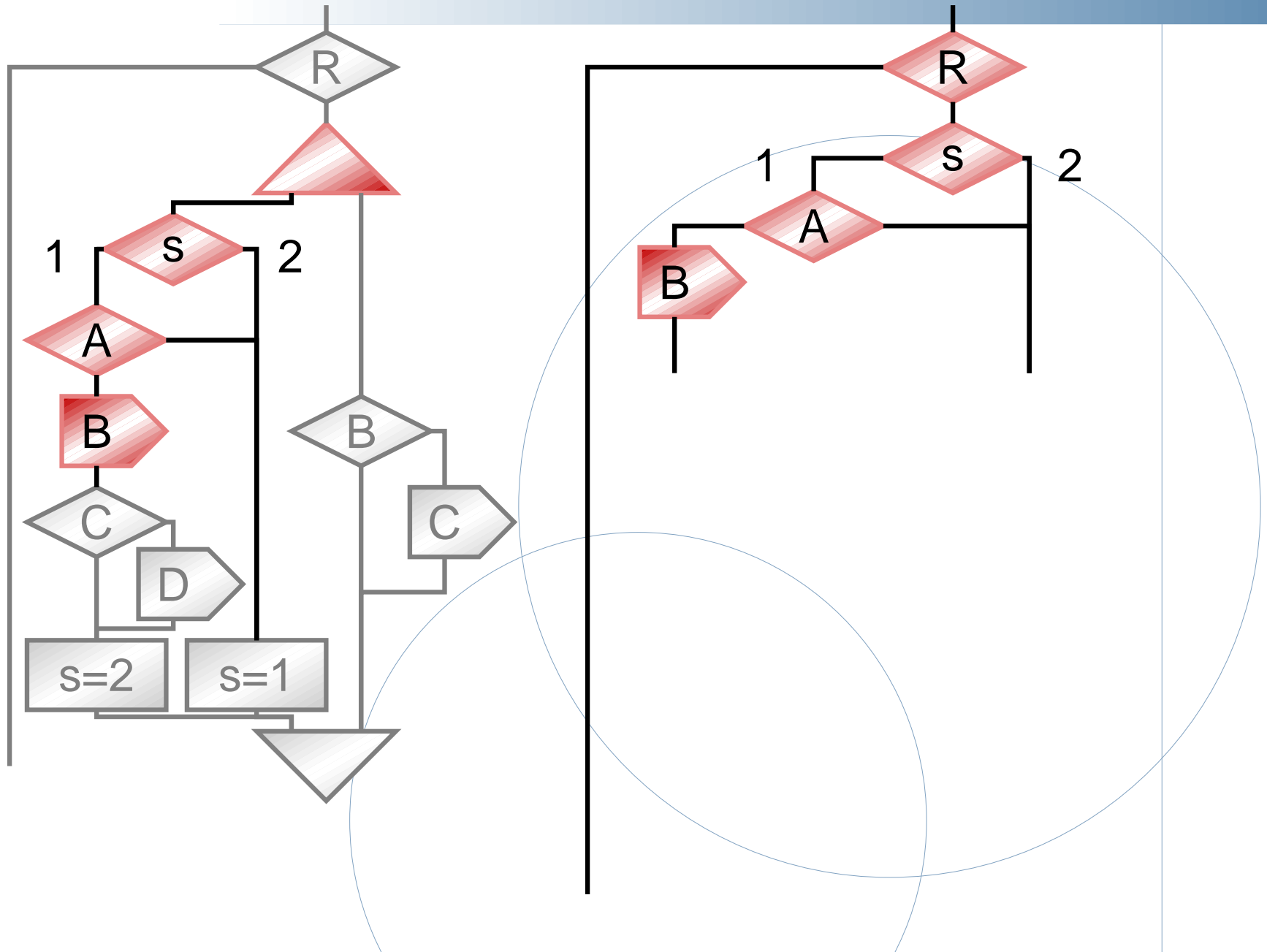
```
every R do
  loop
    await A;
    emit B;
    present C then
      emit D end;
    pause
  end
||
loop
  present B then
    emit C end;
  pause
end
end
```



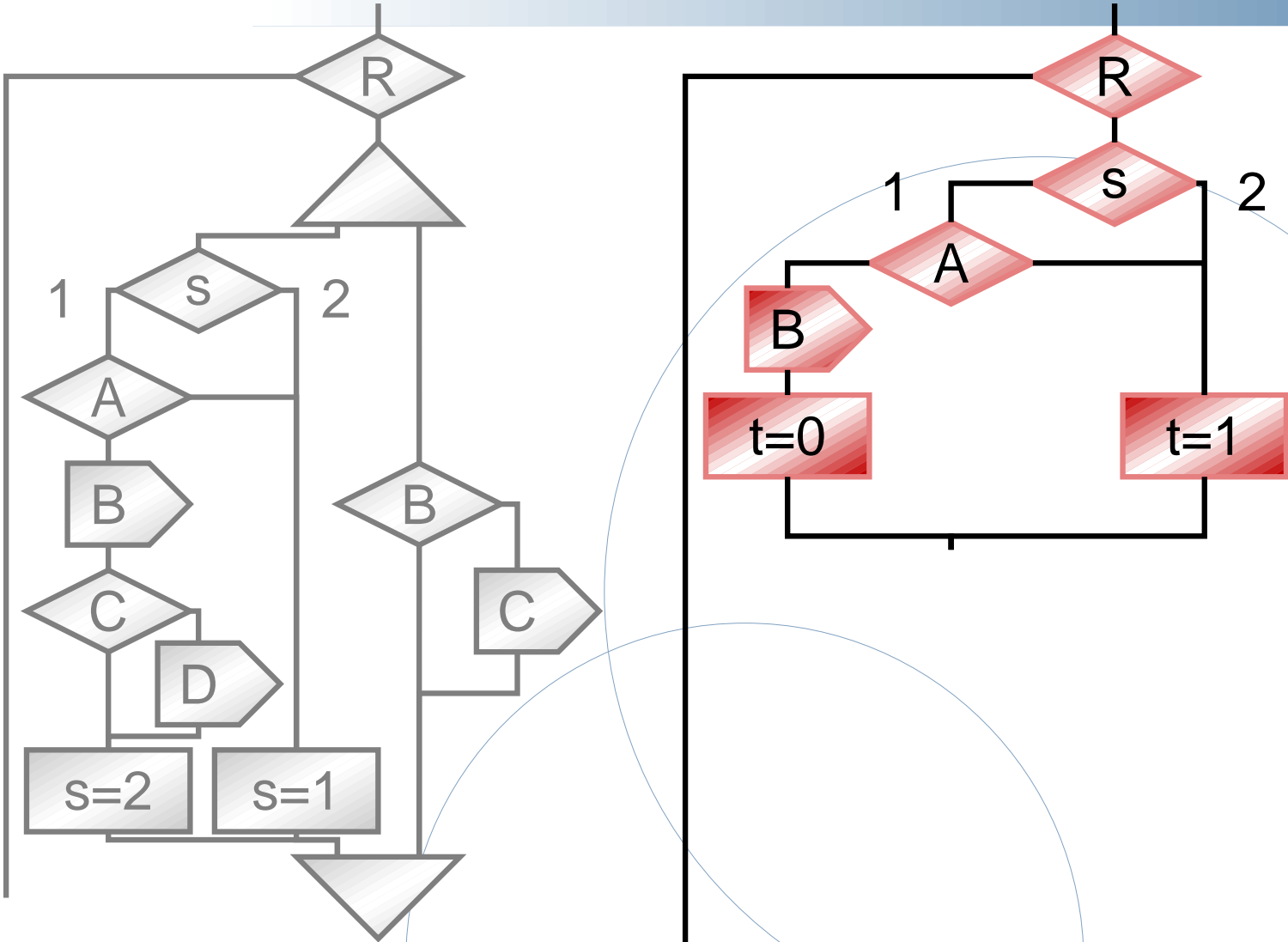
# Run First Node



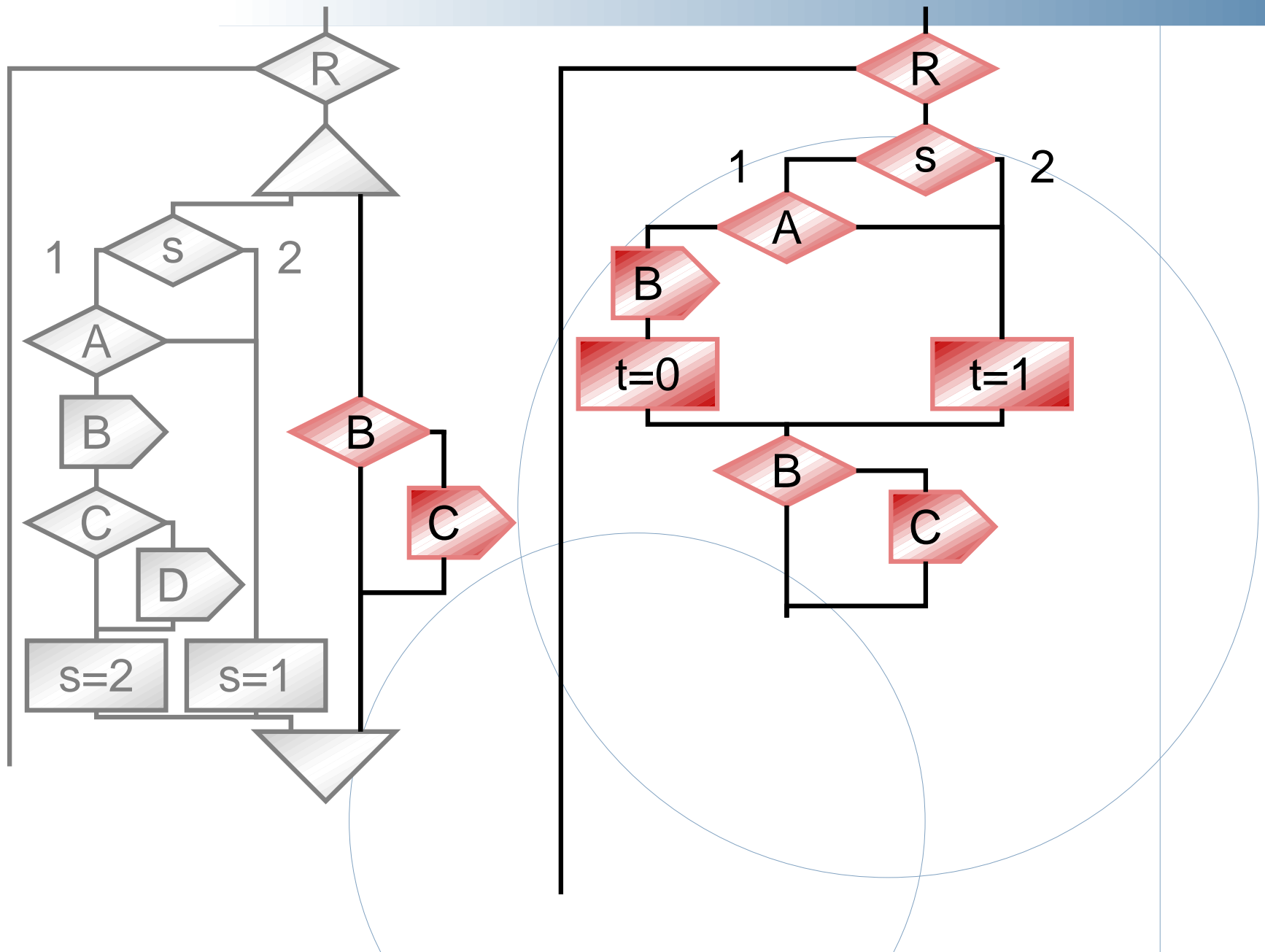
# Run First Part of Left Thread



# Context Switch

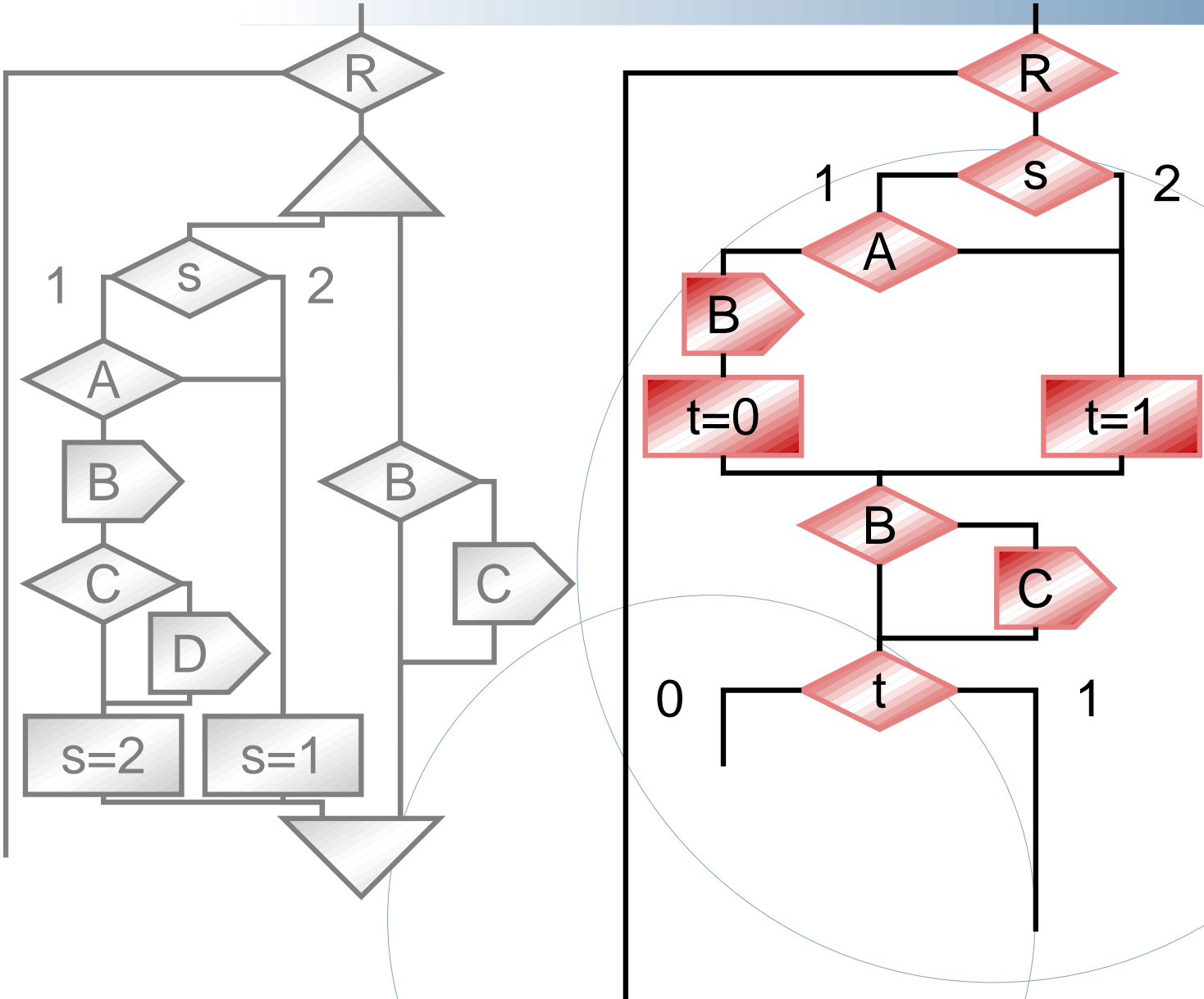


# Run Right Thread

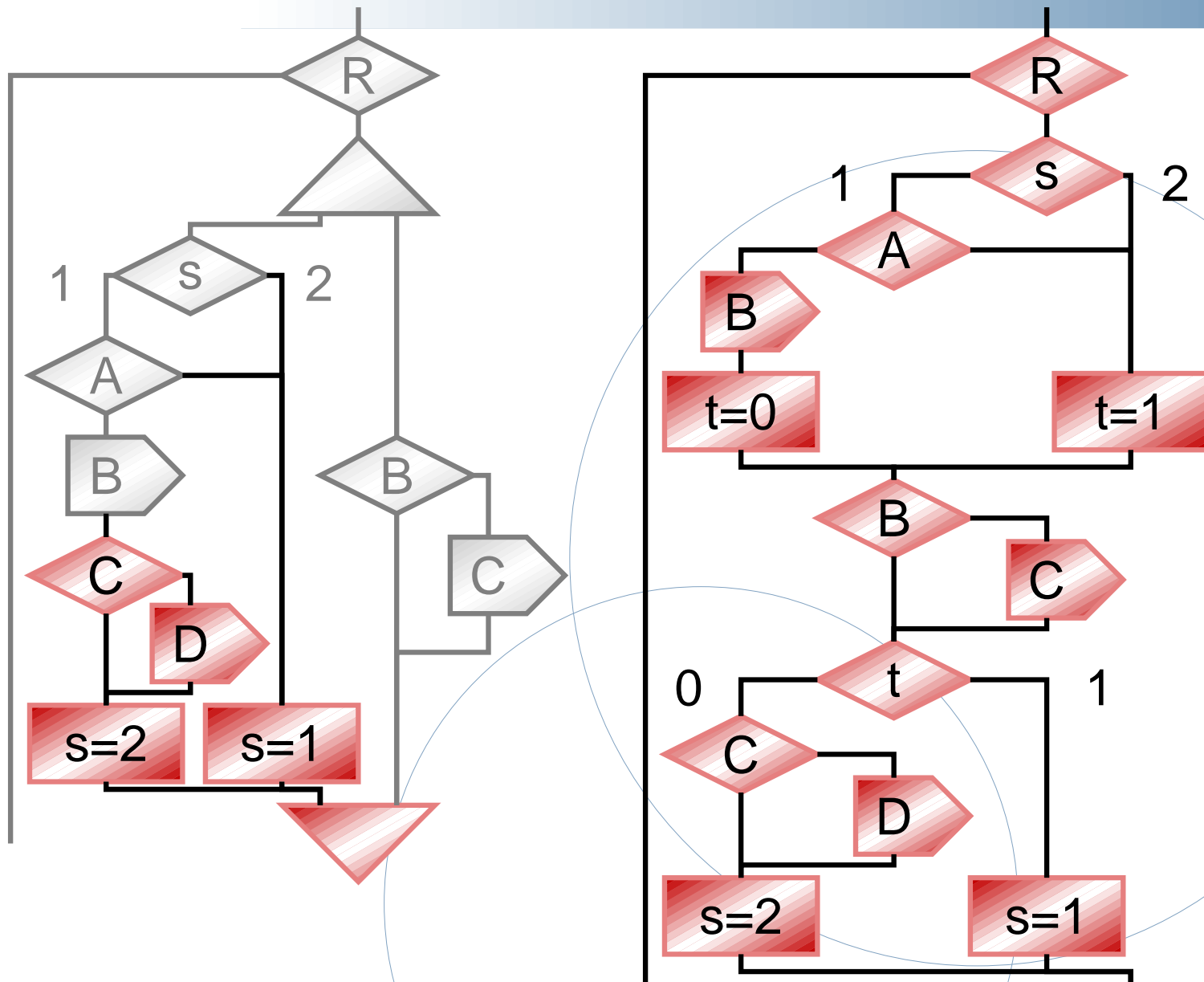




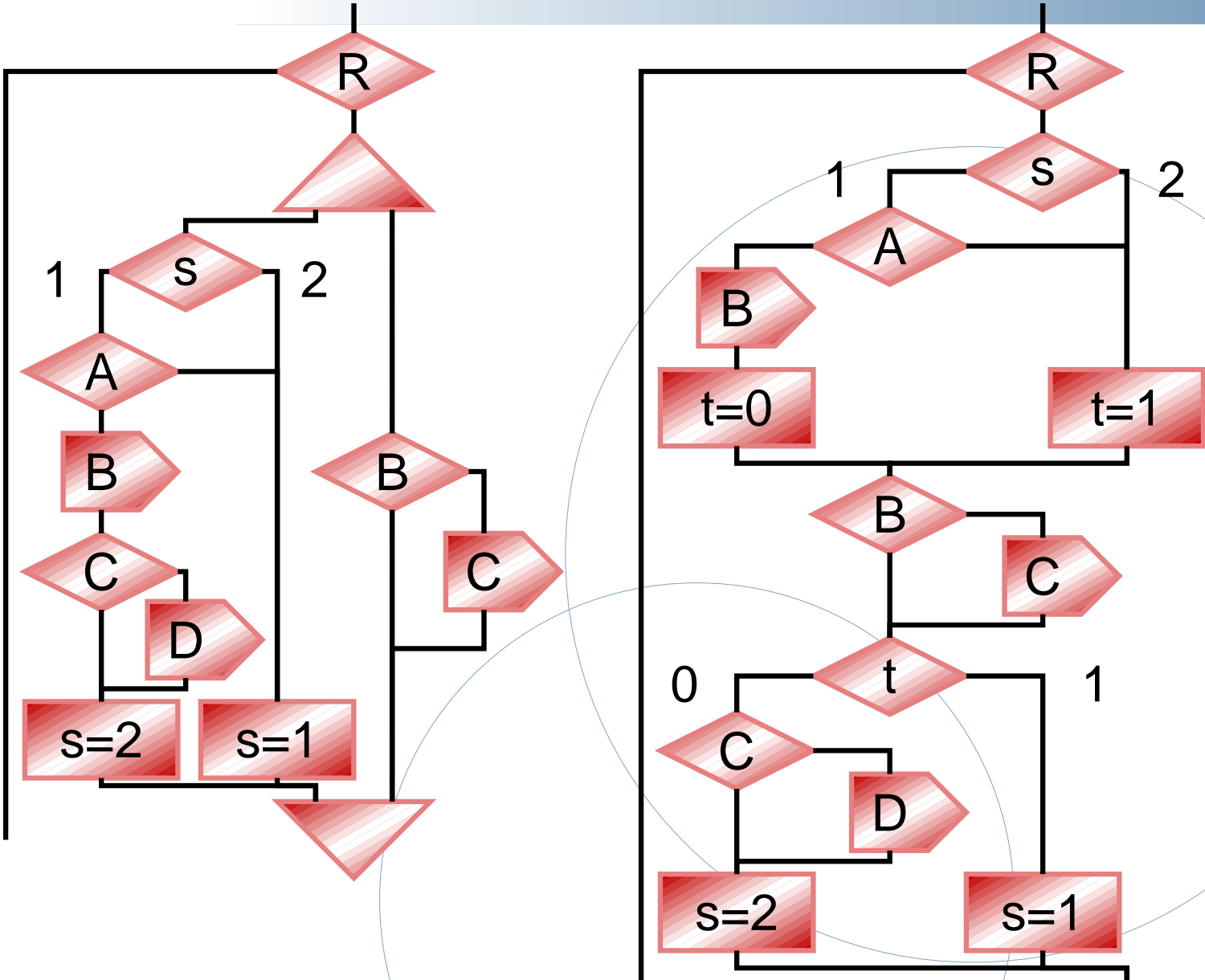
# Context Switch



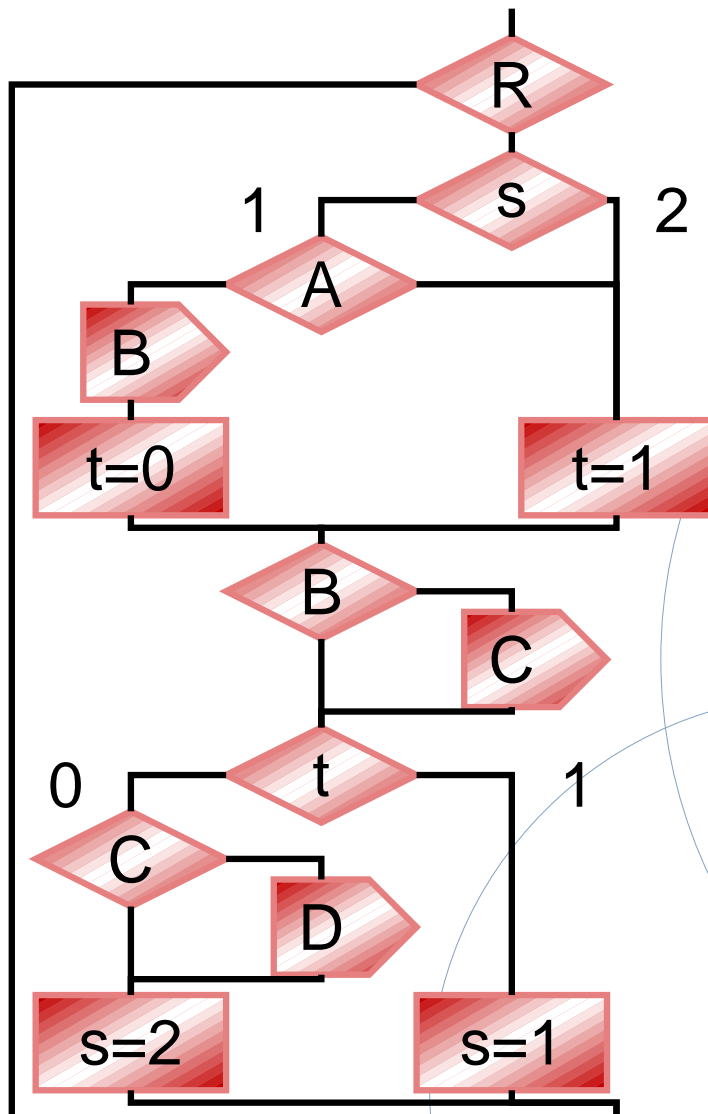
# Finish Left Thread



# Completed Example



# Generated Code



```
if (!R) {  
    if (s == 1 && A) {  
        B = 1;  
        t = 0;  
    } else {  
        t = 1;  
    }  
    if (B) C = 1;  
    if (t == 0) {  
        if (C) D = 1;  
        s = 2;  
    } else {  
        s = 1  
    }  
}
```

# Summary

Plummeting transistor cost is making it practical to put more, smaller computer systems everywhere.

Implemented with SoC technology, these embedded systems will be dominated by software.

Embedded system challenges:

- Real-time issues
- Concurrency
- Software complexity and reliability

# Summary

Esterel and the synchronous paradigm solve some problems

- Synchronous model provides deterministic concurrency
- Finite state permits automatic model checking
- Execution time verification provides timing assurance
- Efficient compilation scheme eliminates OS overhead