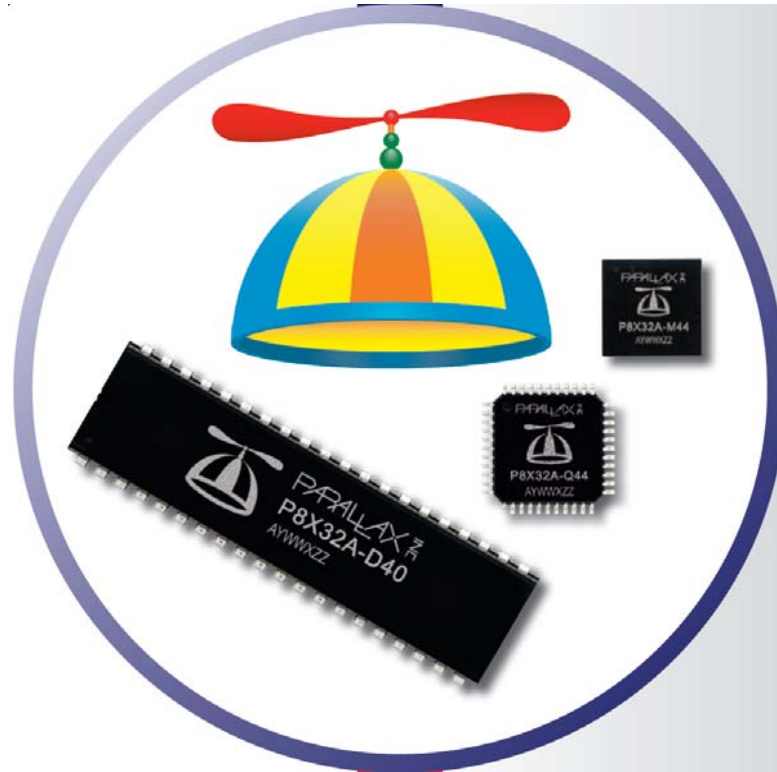


Spin Tips

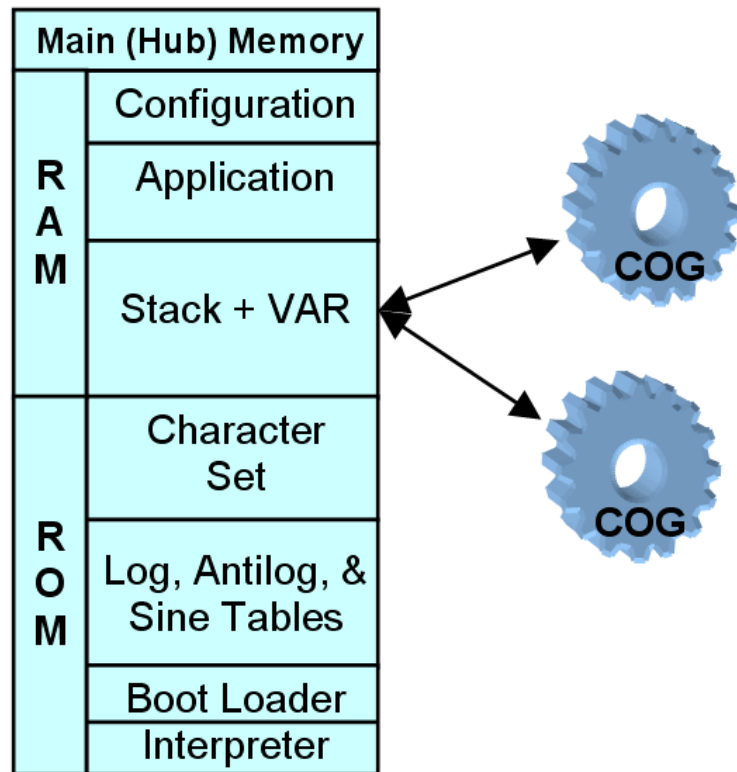


Lock Bits

for coordinating cog access to
shared variables



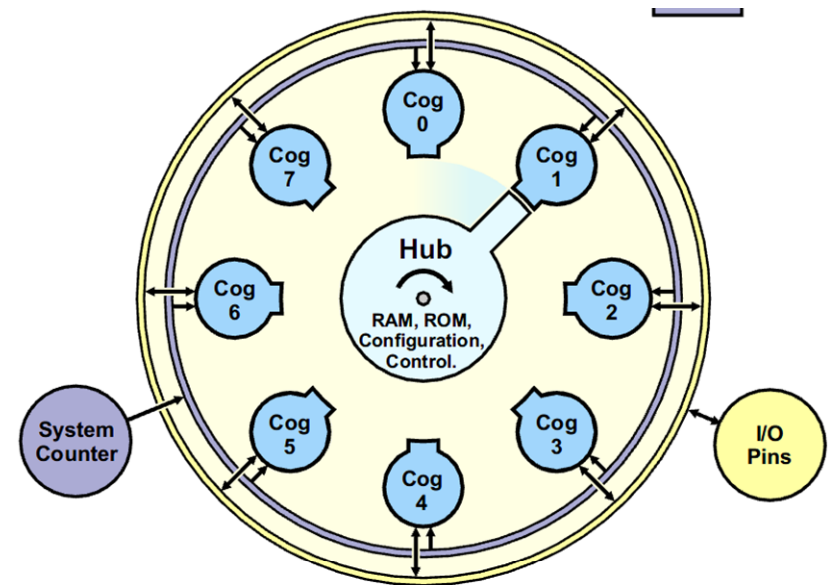
Cogs exchanging information



Cogs exchange values by Main RAM reads and writes

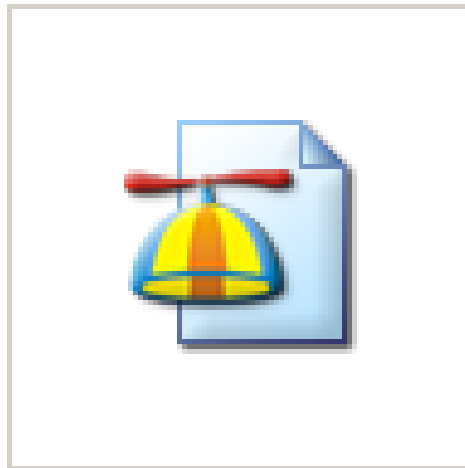
One Value? No Problem

- Data collisions not possible for primitives (long, word, byte)



Example Program

- Remember last month, when we used global variables to exchange information?
- Example program uses only one global variable for information exchange between cogs, so no problem!

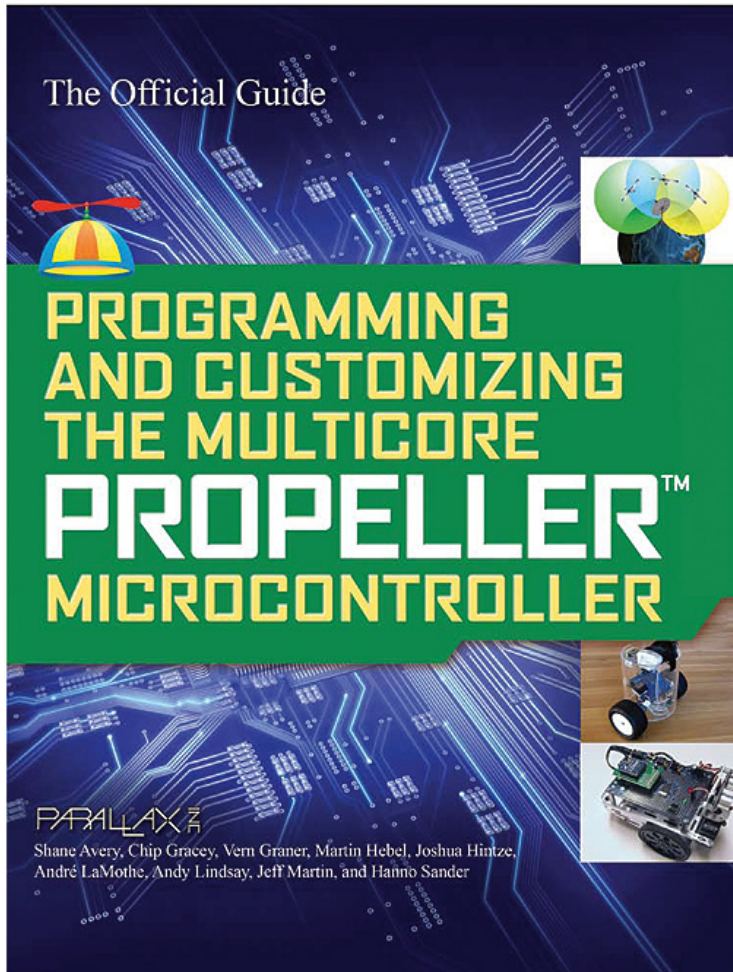


03 Launch Process in
New Cog.spin



Example from the “MGH Book”

www.parallax.com item code 32316



- 1 Propeller Overview
- 2 Intro to Programming
- 3 Debugging Techniques
- 4 Sensor Basics
- 5 Wireless (XBee networks)
- 6 Balancing Robot
- 7 Robot Vision
- 8 Network Apps
- 9 GPS Datalogger
- 10 Propeller Media App
Peripheral
- 11 HVAC Green House Model
- 12 Speech Synthesis

Book is for sale only, but **all example programs**
from this book **free downloads** at:

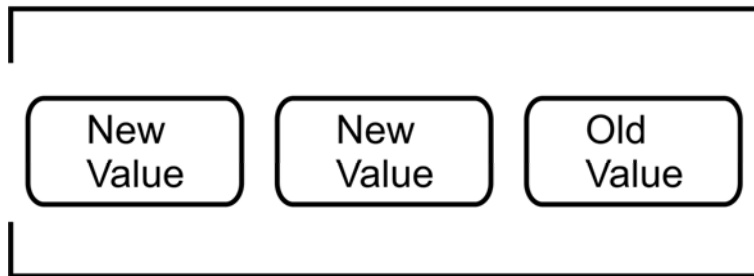
ftp://ftp.propeller-chip.com/



So, what's the problem?

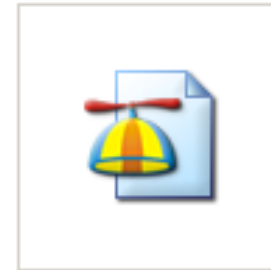
Example

One cog has updated two of the three values.



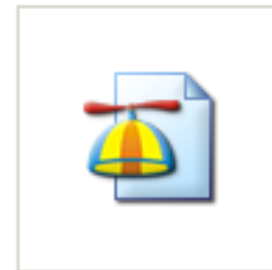
Another cog fetches all three values, two new and one old.

- The problem can go unseen.



Test_Timestamp_from_Another_Cog.spin

- So, testing boundary conditions is very important.



Find_Hidden_Bug_in_Timestamp.spin



Source: MGH book pages 91-102

Lock Bits for Mutex

- What is a Mutex?
 - Short for mutual exclusion, it's shorthand for an algorithm to prevent multiple processes from simultaneously accessing the same memory resource.
 - Long answer:
http://en.wikipedia.org/wiki/Mutual_exclusion
- Does Spin have built-in Mutex support?
 - Yes, Locks.
 - That's eight bits and four commands: LOCKNEW, LOCKSET, LOCKCLR, & LOCKRET.
- This example is based on pages 91-102 in the MGH book.
- Reference material for this example can be found in the Propeller Manual's sections on:
 - **Locks**
 - **LOCKNEW**
 - **LOCKSET**
 - **LOCKCLR**
 - **LOCKRET**
- Another excellent lock resource
 - The Propeller Q&A
 - <http://www.parallax.com/portals/0/propellerqna/>



How to Use Locks

- Step 1: Check out a lock bit with LOCKNEW & store in var (like semID).
- Step 2: Make sure nobody else is using the memory by repeatedly calling LOCKSET until it returns zero.
- Step 3: Work on group of variables that's protected by locks. (Other code that works on the same memory also has to use Step 2.)
- Step 4: Call LOCKCLR as soon as you're done with the group of variables.
- Step 5: If the process is done with the lock, return it! (With LOCKRET)

```
if (semID := locknew) == -1  
  debug.Str(String("Error, no locks!"))
```

```
repeat until not lockset(semID)
```

```
longmove(e.minutes, em, 3)
```

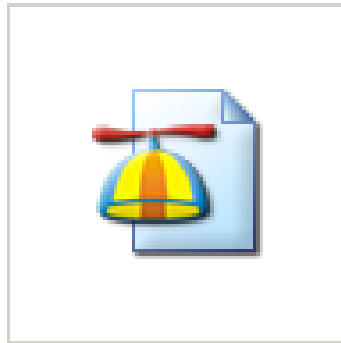
```
lockclr(semID)
```

```
lockret(semID)
```



Example Programs

- Test the repair.
- In this example, pay close attention, and you'll see when the lock had to wait for the values to update.

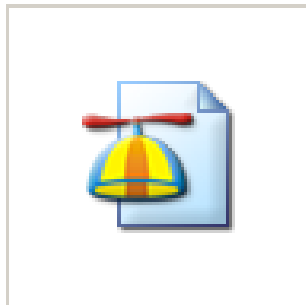


Test_Timestamp_Bug_Fi
x.spin

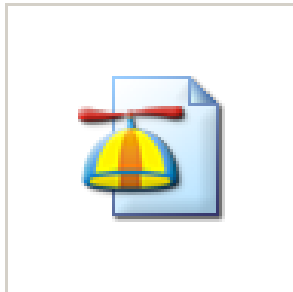


Example Programs

- You can also use it in library objects.
- The library object provides the interface, and both the process and the Method or ASM and the Public methods can use the locks.



Test_TimeStamp_Object.
spin



TimeStamp_Object.spin

