# Embedded and Real Time Software

State Machines

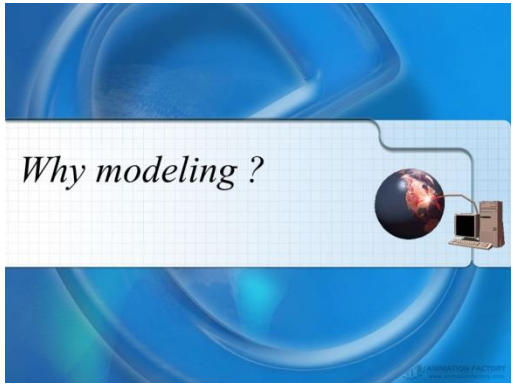
# Readings

- What did you get from the reading?

- http://en.wikipedia.org/wiki/Finite-state_machine

- Tools:
- https://www.iar.com/iar-embedded-workbench/add-ons-and-integrations/visualstate/
- https://www.itemis.com/en/yakindu/state-machine/
- https://www.altova.com/umodel/state-machine-diagrams
- https://www.st.com/en/embedded-software/fsm-examples.html

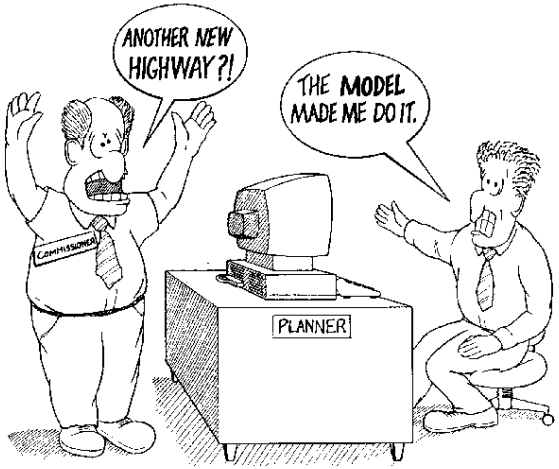
# Why Modeling




- Specifying system behavior
  - Understanding what the system should do
    - In response to external events
    - In response to internal events
- Designing the system
  - Requirements, specifications
  - Understanding odd conditions

# Why Modeling

- Proving properties of the system
  - Proving a system is “safe”
    - What does safety mean
    - Show the system fits this meaning
    - Can be done in terms of models
    - Also can be done in terms of code
  - Performance properties
    - Timing requirements



# Proving Program Properties

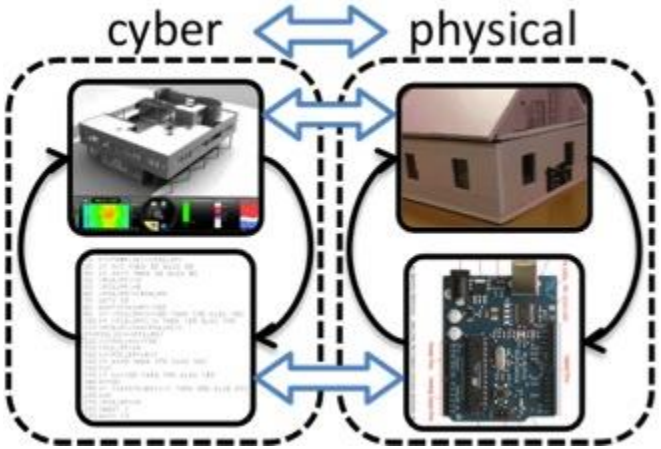
- Prove heat and cold never on within 5 minutes of each other
  - Can do this by looking at the code
    - How complex is the code, how difficult
  - Easier to prove a high-level model satisfies the property
  - And then show the code meets the model

# The Real World

- Embedded systems exist in context
  - That context is the real world
- Is the real world continuous or discrete?

# Physical Systems

- If we embed into a physical system
  - What is the effect
  - What do we want to happen
  - Will we achieve the effect
- Need to model the physical system

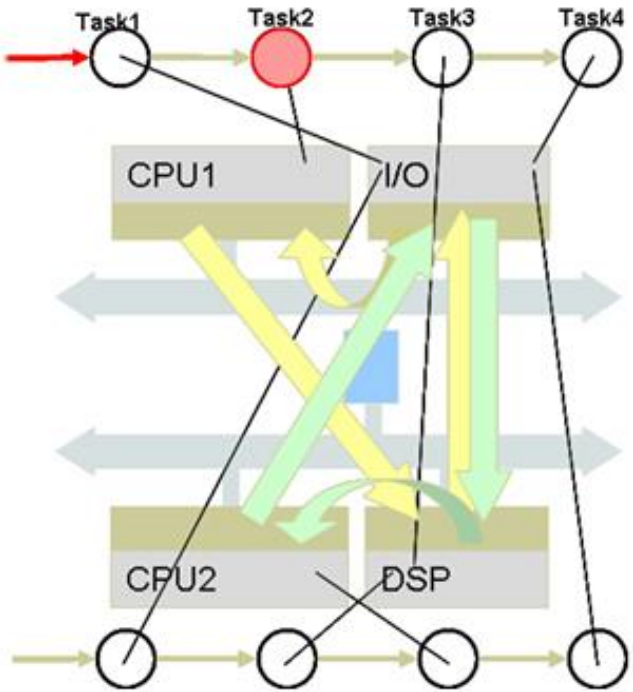


# Modeling for Design

- Why bother with modeling
  - Why not just implement what we want
- What do you want
- How do you know what will work

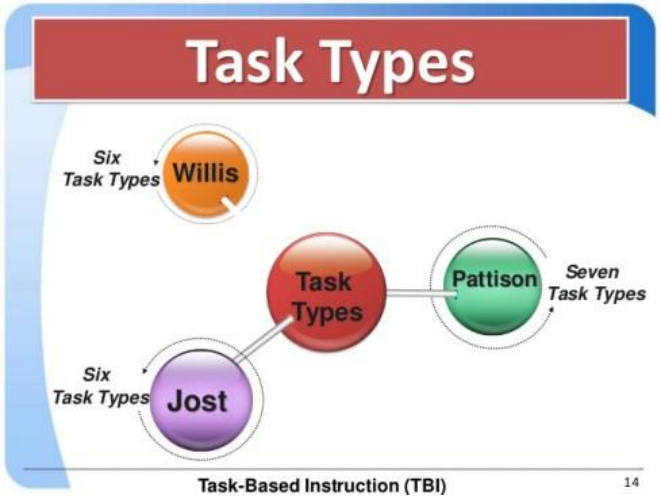
# RT/Embedded SW Architecture

- Break the problem into tasks
- Each task has its own requirements
  - How often to run
  - How much time it takes
  - How critical it is
- What are the tasks for a problem?



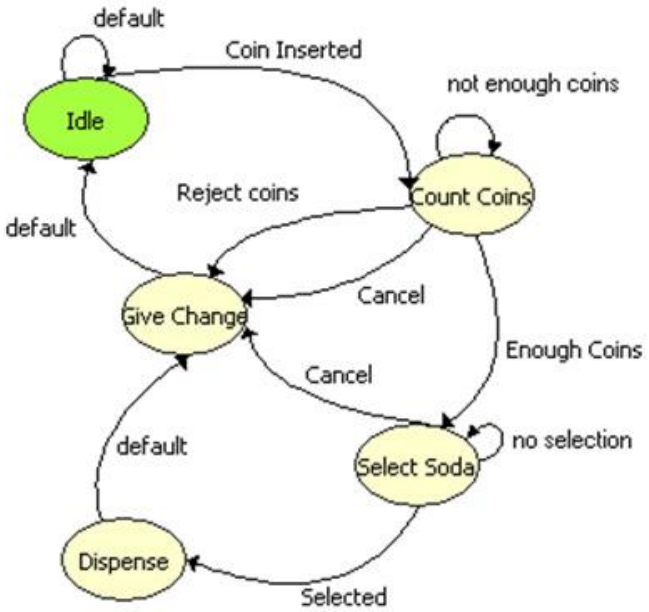
# Task Types

- Control-oriented tasks
  - Managing state, handling sequential actions
- Timer-oriented tasks
  - Something needs to be done periodically
- Data-oriented tasks
  - If multiple tasks access a data structure
  - Have a task in charge of that structure
- Device-oriented tasks
  - Single task to handle a device

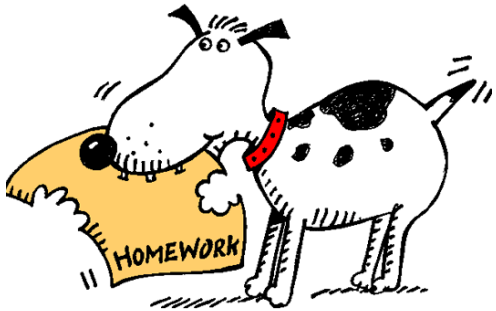


# Modeling the System

- First modeling the individual tasks
  - Interactions with the physical world
  - Interactions based on time
- Then modeling their interaction
  - With each other

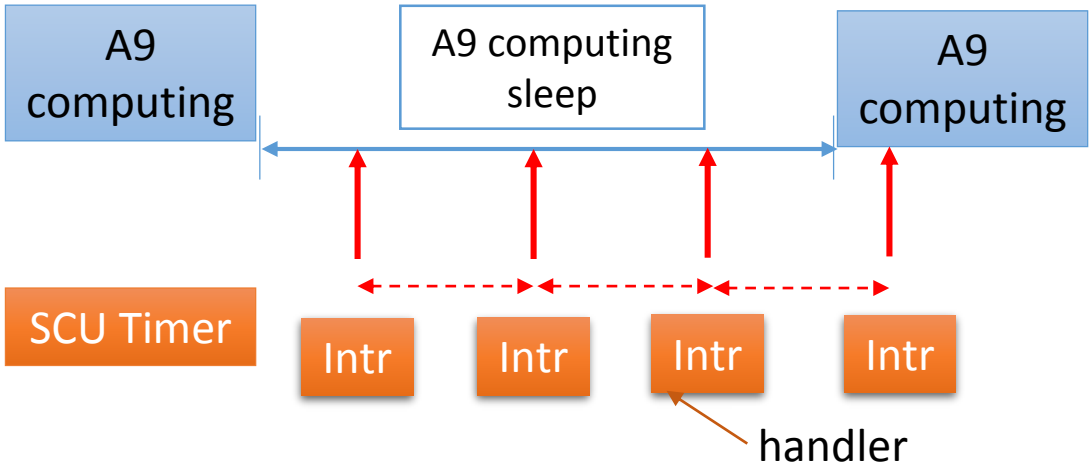


# Homework



# Hardware Timer vs Sleep

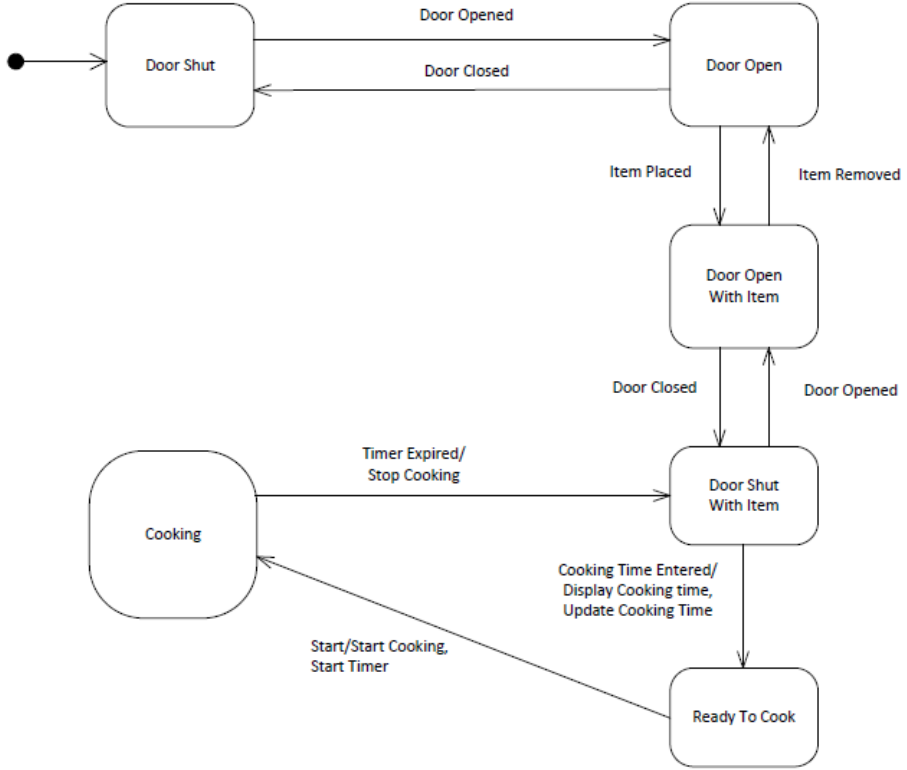
- Sleep(), usleep() : A9 χρησιμοποιεί GlobalTimer / TTC, do…while loop
- SCU timer: 1 per core



# State Machines

- Αφηρημένη αναπαράσταση μηχανής με ορισμένες καταστάσεις
  - Περιγράφει την δομή και συμπεριφορά ενός συστήματος
- Γραφική αναπαράσταση: συνδέει καταστάσεις και γεγονότα
- Event: προκαλεί την αλλαγή κατάστασης
- State:
  - Μια κατάσταση-στιγμιότυπο συμπεριφοράς συστήματος
  - Παραμένει σταθερή για ένα ορισμένο χρονικό διάστημα
  - Το χρονικό διάστημα μπορεί να ορίζεται μεταξύ δύο events
- Action: αποτέλεσμα της εισόδου ή εξόδου σε μια κατάσταση ή της μετάβασης σε επόμενη κατάσταση

# Φούρνος Μικροκυμάτων



# Elevator

- Ιεραρχική State Machine