# CSCE 4114
# Embedded Systems

## Requirements Analysis

David Andrews
Rm 527 JBHT
dandrews@uark.edu

# What We Will Cover Today

- Understanding Requirements
  - Functional
  - Behavioral
  - Timing
  - Physical
- Perform Requirements Analysis on Specific Example
  - Taking Verbal Description and Generating Requirements

# Requirements

Lets Take a Closer Look at Requirements

# Types of Requirements

**Functional Requirements**

- "Depressing button X shall illuminate light Y within 200 msec."

**Non-functional Requirements**

- "Mean time between unsafe operating situations for each train car shall be greater than 250,000 years" (BART specification)
- "Mean time to repair a single component failure shall be less than 30 minutes after arrival of mechanic bringing standard tool set." (typical jet aircraft)
- "Vehicle shall exceed 25 mpg"
- "Elevator system shall deliver at least 1000 passenger*floors per minute at up-peak"

**Constraints**

- Specifies a required technical or other approach
  - "CORBA technology shall be used"

Specifies regulatory or other constraints on solution space/design process

- "System shall conform to requirements of DO-178b" (FAA software process)

# Detailed Behavioral Requirements

1. **List subsystems**
   - In a fine-grain distributed system, this is sensors+actuators+objects
     - "Other objects" are usually compute-nodes such as dispatcher
   - Actors included to provide environmental model and interface
2. **Create a database of behavioral requirements for each subsystem**
   - Replication
   - Instantiation
     - Initial conditions
     - When are dynamic objects are created?
   - I/O interface (list of sensor/actuator interfaces)
   - State (private variables useful for describing behavioral memory)
   - Constraints
     - Non-functional requirements; safety interlocks
   - Behaviors
     - Functional requirements

# Some Characteristics of a Good Requirement

**Understandable and unambiguous**

- Understandable to system designers who have to implement it
- Understandable to marketing/sales/customers who know the needs
- Understandable to outside vendors (non-domain experts) who have to build it
- Concise (few words) and simple

**Implementation-neutral**

- Doesn't say how to do it, just what the desired outcome/action/side-effect is

**Testable**

- If you can't test it, how do you know the system meets the requirement?
  - Constraints can be difficult – how do you prove something is impossible?

**Traceable**

- Must be possible to trace that requirement through final system
  - Sometimes done by tracing to features/system component functions
  - More often done in practice by tracing to system tests

# Modeling The System

**Build models to check requirements**

- Making model executable forces designer to confront details
- Models help look for bottlenecks and explore design alternatives

**We're going to discuss common examples of different models**

- "Plumbing diagram" for capacity analysis
- Statistical/queuing models
- Functional models
- Analytic models
- Hybrids

**Time can be treated as: Discrete Event / Continuous / Cycle / Hybrid**

# Capacity Models (with network example)

"Plumbing Diagram" / spreadsheet:

- Messages are delivered from source to destination
- Example: Maximum bandwidth 1000 msgs/sec (might also be in bytes/sec)
  - But nodes may be limited by CPU power to lower bandwidths

# Statistical Models

**Statistical:**

- Maximum bandwidth
- Time delay at 50% of maximum bandwidth
- Ignore noise, failures

**Detailed statistical:**

- Delivery time (including retries)
- Loss probability (with limited retries)

- Maximum bandwidth possible
  - Service degradation *vs.* used bandwidth

**Queueing theory**

- (Math beyond the scope of this course…)

# Functional & Implementation Models

**Functional**

- CAN network is a priority queue
- Feed messages to network and see what happens

**Implementation**

- Verilog for CAN chip implementation…

**But, some things usually don't matter**

- Implementation below level of abstraction
  - *e.g.*, if motor is controlled with voltage, current, pulse-width modulation
  - *e.g.*, type of sensor/actuator as long as it implements the correct function

# Simulation Models

**Coupled analytic models *(e.g.,* spreadsheets)**

- 12 messages, 113 bits each, sent at 20 Hz
- 5 messages, 118 bits each, sent at 2 Hz ...

**Continuous Time models**

- Usually used in physical domain *(e.g.*, computational fluid dynamics)

**Cycle-based models**

- *e.g.,* Verilog
- Ends up being how time-triggered simulations run
- Simulation speed proportional to length of time simulated

**Discrete Event simulations**

- Events are queued for action at specified times *(e.g.*, wait 100 msec then do X)
- Simulation "clock" fast-forwarded to next queued event at zero cost
- Simulation speed proportional to number of events

# Generating Requirements

- Now
  - We will generate requirements for a Flight Controller
  - I am the customer
  - You are the Engineers/Scientists…..
  - Lets Play…..

## Requirements Analysis

- How do we determine requirements ? We study the written statement of work, and ask the rep questions.....
  - Questions to Ask
    - What does it do ?
    - What is it interfacing to ?
    - How fast does it do it?
    - How big can it be?
    - How much power can it draw, and produce?
    - Other special requirements ?
- You must understand requirements in order to design system

## Razor_Hawg Flight Controller

- Scenerio: Cessna comes to you and wants you to design the autopilot system for there new plane, the Razor_Hawg Commuter Airplane.

- Step 1. Determine system requirements, then provide a requirements document to the rep. We will not "cost" it out. If I agree to what it is you are going to build, your finance department will work up the cost (obvioulsy this is an academic excersize, not the real world) and submit proposal for us.

  - The requirements document will be your agreement with me.

## What does it do ?

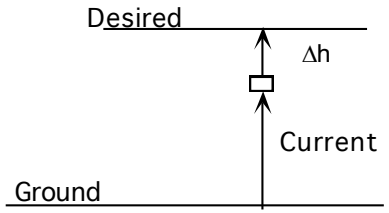
- Question: What does it do ?
- My Answer: Auto pilot steers the airplane.

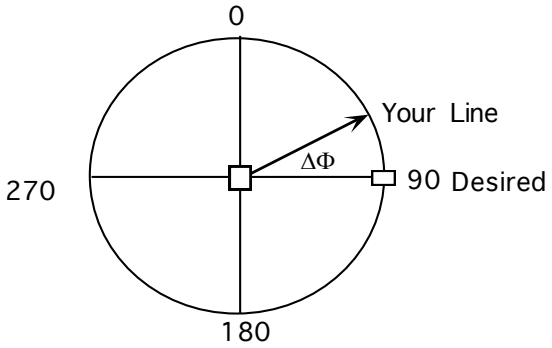
- Question: How does it do this ?
- My Answer: Takes input signals from sensors, computes adjustment factors, computes new steering vector based on adjustments and updates controls.

- Question: What are interface signals and how does it compute adjustments
- My Answer: Let me draw a picture.....

## System Inputs

- 2 Variables {Absolute Azimuth, Elevation}
- Elevation Azimuth



## Interfacing
## More Position Information

- Flight sensors will provide digital input values Specifying Current Position

- Resolution of Flight Sensor (Inputs) are:

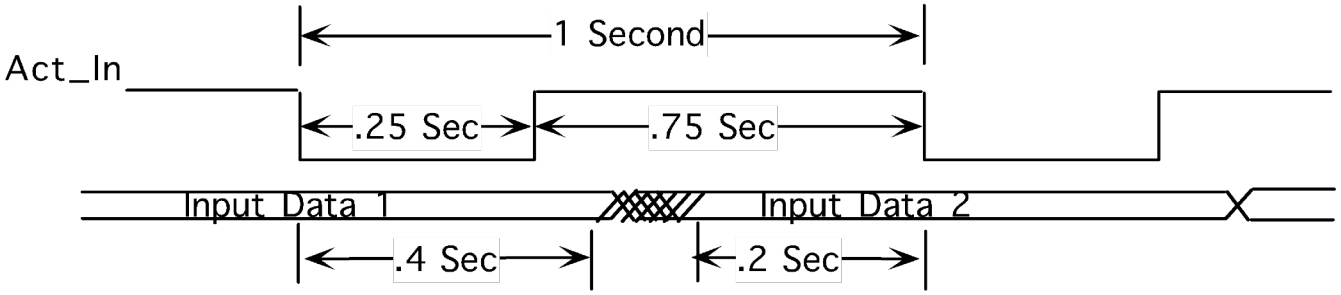
| | | |
|---|---|---|
| Position | 1.5 degrees 360 total. 240 values | -> 8 bit A/D |
| Height | 50 ft. sea Lvl to 12000 ft. 240 values | -> 8 bit A/D |

- Steering adjustments (Outputs) are:

| | | | |
|---|---|---|---|
| Position | -25 < desired < +25 | 1 degree Incrts | 6 bit A/D |
| Height | -100 < desired < +100 | 1 ft Inctrs | 8 bit A/D |

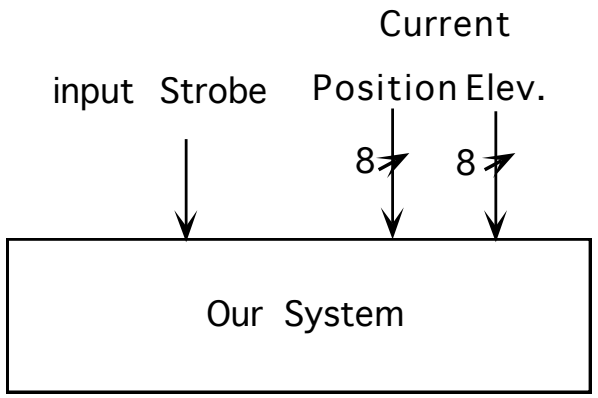
# How Do We Know When New Inputs Are Available?

- Will Provide Timing Signal On A per Second Basis. Timing Signal Looks Like:



# Block Diagram For Current Position

- Current Position Requires 2 Inputs and a Synchronization Strobe.
- -Number of Inputs Defined
- -Size of Inputs Defined
- -Timing Signal Defined



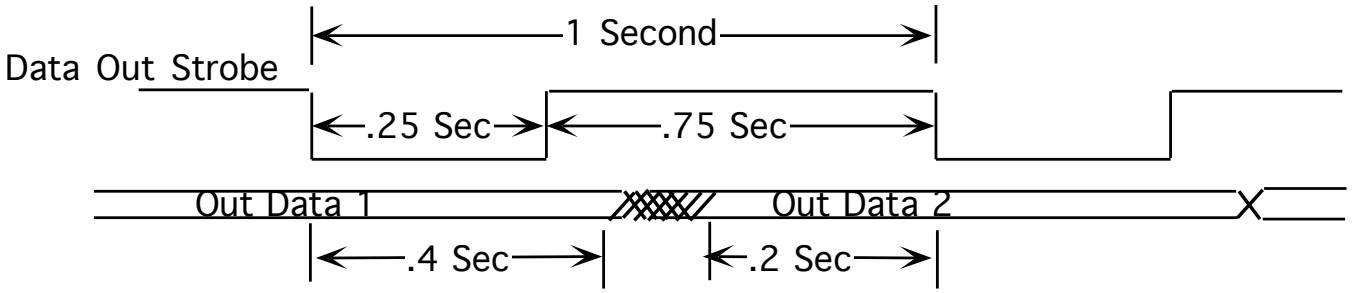
## Output Signals Must Be Defined

- Steering adjustments (Outputs) are:

  Position -25 < desired < +25 1 degree Incrts 6 bit A/D

  Height -100 < desired < +100 1 ft Inctrs 8 bit A/D

  - We Must Provide Two Output Signals, Along With a Synchronization Signal. The Signals Are Δ's That Go To Alerons and Height Controls.

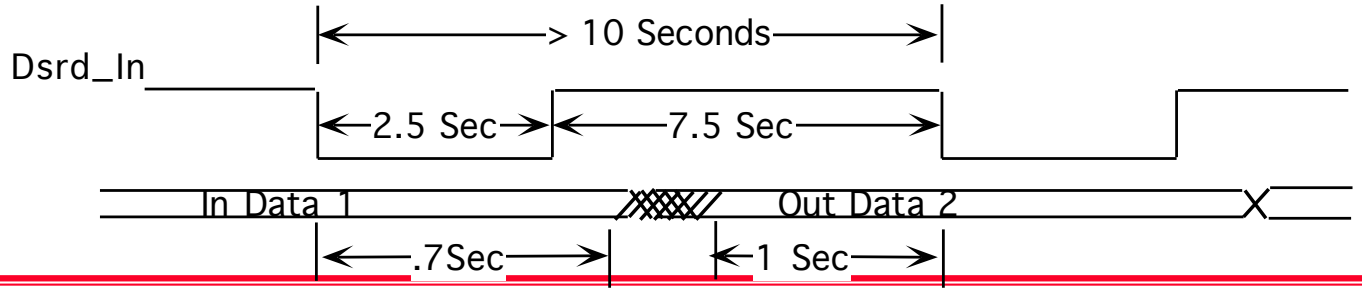


# Desired Position

- Desired Position is "Dialed In" From Pilot. This Happens About 30 Minutes to Over 1 Hour. Desired Position Specifications are:

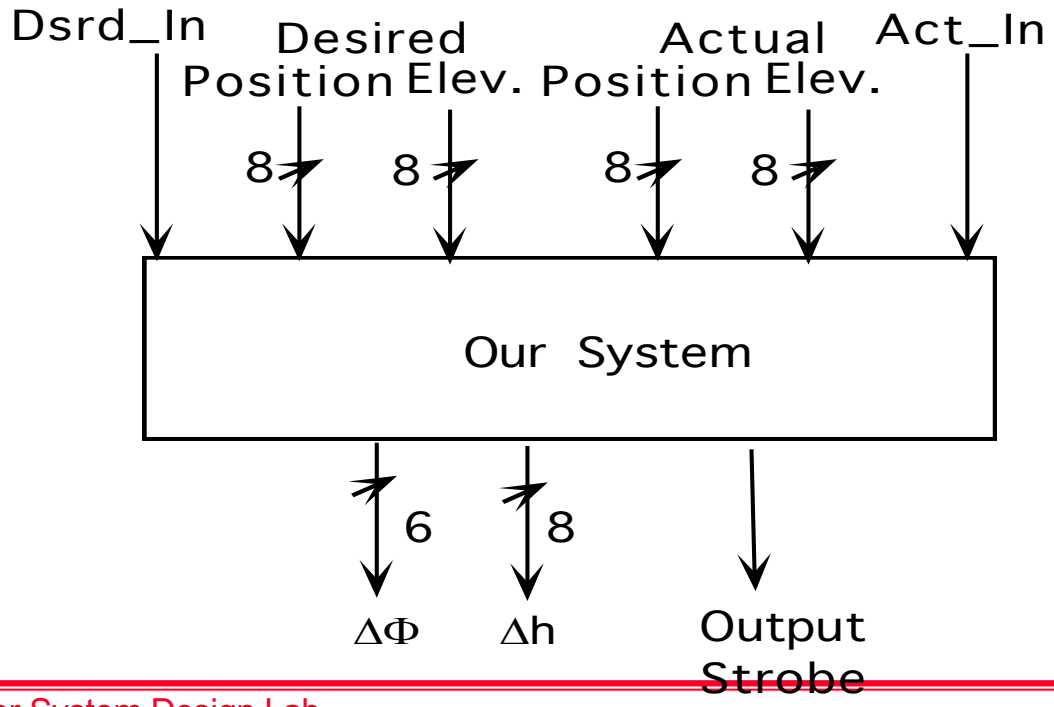
- Resolution of Desired Azimuth and Elevation Same as Current Position
  - Desired Direction 6 bits
  - Desired Elevation 8 bits

- Simple External Input Timing Signal Available: Negative Edge Triggered When Pilot Updates New Desired Position



## Interfacing Descriptions

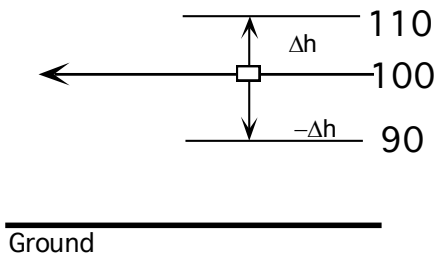
- Updated Block Diagram:
- Inputs and Synchronization Defined
- Output and Synchronization Defined

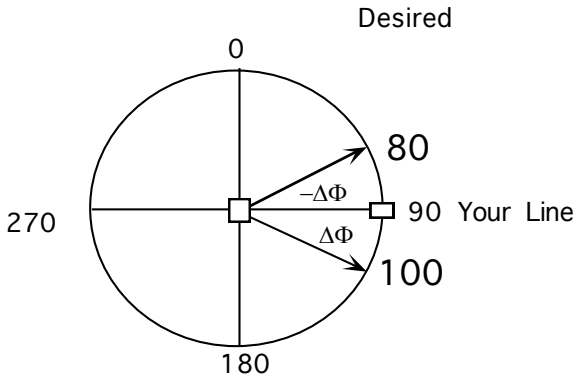


## How Fast is it ?

- Input Samples Will Come in Every Second: Need to Make Compute Adjustments and Output Before Next Sample.
  - Calculations

$\Delta h = h_{desired} - h_{true}$ $\Delta\phi = \phi_{desired} - \phi_{true}$




- $\Delta h = h_{desired} - h_{true}$
- +10 = 110 - 100
- -10 = 90 - 100

$\Delta\phi = \phi_{desired} - \phi_{true}$

-10 = 80 - 90

+10 = 100 - 90

- Calculations simple: 8 bit Integer adds/subtracts

## A little analysis so Far

- We have preliminary information that can help us to derive some requirements

  - CPU Requirements Based on:
  - {Speed, Computations, Memory Requirements}

  - How much memory ?

ROM Holds Program {Initialization, Routines}

RAM Holds Data {Data}

# CPU Requirements

- Operations are integer. Implies We Don't Need to Go To a CPU With Floating Point Capabilities.

- Speed: Updates Every Second. Calculations are Simple:

$$\Delta h = h_{desired} - h_{true}$$

$$\Delta \phi = \phi_{desired} - \phi_{true}$$

- Assume 20 Cycles Per Instruction. This Program Should Require Less Than 2000 Instructions (Best Guess). Therefore

- Cycles per Update = 20 Cycles/Instr * 2000 Instrs = 8000 Cycles/Update
- Frequency of CPU > 40 Khz, No Need For Expensive Pentium !!!

# CPU Requirements Continued

- Total Memory Size is Determined By Address Width of CPU.
- Program Size Will Be Small 2000 Instructions
- Data Storage Will Be Small < 1kbyte of Data

- Data Width Also Determines
- 8 Bit Data No Problem

- A Cheap CPU Looks Ok. Other Things To Consider
  - Development Environments
  - Availability

## CPU / Memory Requirements Summary

- CPU Requirements Driven By:
- Functionality
- Speed
- Addressing Size
- Data Size

- Memory Requirements Driven By:
- Size of Program
- Size of Data

How big can it be?
How much can it weight ?

- Must fit in the following area:
  - Length: 14 in
  - Width: 14 in
  - Heigth: 2 in
- Must weight less than:
  - Weight: 2 lbs

## Power Considerations

- Power supply from plane:  budgeted for +5 volts  3.5 Amperes
- Must operate in temperature range of 0 to 40 C.

## Requirements Summary

- We Know What It Does - Inputs Data, Computes and Outputs 2 Deltas
- Inputs:
  2 8-bit Inputs Actual Position, 2 8-bit Inputs Desired
  Input Sample Period (Actual Pos 1 sec, Desired Pos >10 secs)
- Outputs:
  1 6-bit output, 1 8 bit output
  Output Sample Period 1 Second
- CPU:
  Integer arithmetic, simple operations, 2k Instrs, 1k Data
- Memory:
  RAM, ROM

One More Thing and We Are Done.....

## Special Requirements

- "We Need to Be Able To Debug Boards and Update Algorithms"

- Debug Support
- Make life simple, provide interface for debugging from PC
  - RS 232 Serial Interconnection
  - Firmware for developing/debugging programs

# Final Block Diagram