CSE 466: Course Project

- Complete a large project that embodies the major course topics
- Project should be simple but expandable
- The project should include:
  - Multiple device communication
  - Deal with constrained resources
  - Control hardware by directly manipulating the I/O
  - Introduce an embedded OS
  - Participate in a multi-agent project – team effort
  - Use current technology

The Flock

- Two week project to tie together everything you've learned in 466
  - Programming microcontrollers
  - Wireless radio communication
  - Embedded operating systems
- A piece of "performance art"
- Allows nodes programmed by different students to work together
- Exposes some problems of scale in building sensor networks

Basic Idea of the Flock Project

- Each node ("bird") sings a song
- It listens to its neighbors to hear what they sang
- It makes a decision as to which song to sing next
  - This can lead to an emergent behavior – property of the group
  - We'll be trying for an effect that propagates a song around the flock
- If it is startled (by a shadow cast on its light sensor), then it makes a "scared" noise and informs its neighbors who will do the same
- If it is "selected" (by a repeating shadow on its light sensor), then it send a packet to the controller
- It synchronizes with neighbors by adjusting to time values in every packet it receives
- It responds to commands from controller
  - Adjust parameters
  - Turn on LED
  - Sing a specific song at a specific time
- Feel free to experiment

Flock State Machine

- WAIT STATE (silent, go to this state on reset)
  - Wait to receive a packet of type AdjustGlobals
  - Turn tri-color LED off and turn on red LED on corner of sound board
  - Ignore SangSong packets from neighbors
  - IF (AdjustGlobals) set clock, go to clear state
  - IF (Command Packet) perform command, change LED, possibly sing song, stay in wait state
- CLEAR STATE (silent)
  - Clear correlation FIFO and receive queue data (all historical data)
  - Clear all counter variables
  - Wait for random amount of time (1000- 4000 milliseconds)
  - Go to sing state
- SING STATE
  - Choose a song (using provided algorithm) and send to Yamaha chip
  - Collect data from neighbors even while singing
  - After song is finished, send SangSong message
  - Go to listen state
- LISTEN STATE (silent)
  - Set listem time for random t ϵ [minListen, maxListen] msec
  - Listen and collect data from neighbors
  - When listen timer goes off go to sing state
- STARTLED STATE
  - Send "startled" song to Yamaha chip
  - Collect data from neighbors even while singing
  - Send startled message to neighbors (remember to decrement hop count)
  - Turn tri-color LED off and red LED on corner of sound board on
  - After finished singing, delay 10 secs, turn off red LED
  - Go to listen state

Active Message Types

- SangSong
  - Inform neighbors as to song that was just completed
- AdjustGlobals
  - Change parameters
- StopNWait
  - Go to wait state
- CommandPacket
  - Adjust LED and possible sing song
- Startled
  - From neighbor, indicated scared bird
- Selected
  - Send back to controller when selected

Song Decision Algorithm

- Goals
  - Sing the same song for a little while
  - Songs start, then spread, then die out
  - Don't sing the same song too often
- Algorithm
  - Determine nearest songs
  - If our song = any of nearest n, then repeat song
  - If all same, switch to different song
  - If none same, switch to different song
  - If selected song on "black list" pick a different song
- How do we evaluate how effective this algorithm is?

Song Decision Algorithm (cont'd)

```
x = rand() % Probability
y = rand() % Silence

if (x == 0)
  SONG = song with the lowest point value
else if (y == 0)
  Silence, don't sing a song, go back to LISTEN STATE
else
  SONG = song with the highest point value
```


Synchronizing Time

- All packets include timer value
- Take average of local time and received time and adjust timer
- Over many packets this will cause nodes to converge to same time
- How accurate is this approach?
  - What are the sources of latency?
- Can always reinitialize nodes using packet from controller

Selection

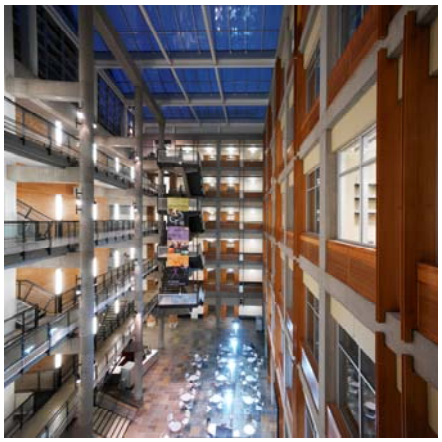
- Command packets include a range of node IDs (if min=max, then a specific node)
- All selected nodes execute command
- Node can identify itself through identification sequence
  - Series of 3 light-dark-light transition on photo sensors
  - Need to be distinguished from single transistion “startle”

Additional Details

- Packets from Node 0 must be specially treated – they may contain global parameters
- Arriving packets must be strength-stamped for RSSI value – special radio stack required

The Concert – Mar 10 – 12:30PM

- Final demo for the class is a concert
- Each student has a mote to contribute (23 motes)
- Same specification but different code in each mote
- The motes have to "qualify"
  - We will have testing scripts to simulate the flock and eliminate nodes that may cause problems
  - Used for grading projects



Conducting

- Nodes will be placed around atrium
- Controller will reset all nodes
- Flock songs will play
- Specific node will be identified
  - Nodes on one side will be told to sing one song
  - Nodes on the other a different one
- Evaluate
  - Selection method
  - Ability to sing songs synchronously