SONOS COM.

CE Presentation

UCSB Capstone Team
Our Team

**MEs:** Kyle Li, Yang Xue, Kenny Wang, Kayden Sung, Yubin Liu

**EEs:** Yiqin Wang, Luke Buckle, Jianyang Lu

**CEs:** Subho Choudhury, Richard Wei, Mohammad Cazi, Brian Sandler, Brenden Fujishige, Marcellis Carr-Barfield

**UCSB**

Tyler Susko, Carl Meinhart, Ted Bennett, Steve Laguette, Trevor Marks, John Johnson, Yogananda Isukapalli, Ilan Ben-Yaacov, Ekta Prashnani, Sean Mackenzie, Caio Motta, Celeste Bean

**SONOS**

Camille Zaba, Nathan Pike, Farhad Mirbod, Daniel Huthsing, Vicki Chen, Gregorio Teller

**Laritech**

Bill Larrick, Veronica Ellias, Lillian Ware, Kristin Bradley

Sponsored by: **SONOS, Laritech**
Our Task

To design and build a convenient communication device that works seamlessly with your existing Sonos systems.
Overview
COM. is an intercom device that can connect and control all SONOS devices in a home network.
Communication
&
Music Control
Capacitive Touch

Living Room
Modes

- Music
- Intercom
Music Control
Intercom
User Setup Procedure

1. Power on COM.

2. Connect to WiFi network “SONOS COM.”

3. Use app to send SSID and Password.
Design and Size Constraints
Competitor Analysis

Amazon Echo Dot

- **Advantages**
  - Multiple functions
  - Smart Controls
  - Low price ($49.99)

- **Disadvantages**
  - Does not have a screen
  - Too many buttons
  - Does not have an intercom function

<table>
<thead>
<tr>
<th>Specification</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weight</td>
<td>5.7 oz (163 grams)</td>
</tr>
<tr>
<td>Size</td>
<td>1.3” x 3.3” x 3.3” (32 mm x 84 mm x 84 mm)</td>
</tr>
<tr>
<td>Connectivity</td>
<td>Wifi</td>
</tr>
<tr>
<td>Power Source</td>
<td>Micro USB</td>
</tr>
</tbody>
</table>
Competitor Analysis

Senic Nuimo

- **Advantages**
  - Premium design
  - Rotation controls
  - Dot display
  - Can be wall mounted

- **Disadvantages**
  - Only controls music
  - Only controls single device
  - Expensive ($199.99)

<table>
<thead>
<tr>
<th>Weight</th>
<th>8.9 oz (254.5 grams)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Size</td>
<td>2.75” (70 mm) Diameter, 0.6” (15 mm) Height</td>
</tr>
<tr>
<td>Connectivity</td>
<td>Bluetooth LE</td>
</tr>
<tr>
<td>Power Source</td>
<td>Rechargeable Battery (4 months of charge)</td>
</tr>
</tbody>
</table>
Original Expected Design Specifications

Design Requirements:
Must Design for Manufacturing, Assembly, Mass Production (Injection Molding), and for Experimentation.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Size</td>
<td>1.5” x 4.25” x 4.25” (38.1mm x 108mm x 108mm)</td>
</tr>
<tr>
<td>Weight</td>
<td>184 grams</td>
</tr>
<tr>
<td>Screen</td>
<td>2.2” (38 mm) Color TFT LCD Display</td>
</tr>
<tr>
<td>Material</td>
<td>ABS Plastic</td>
</tr>
<tr>
<td>Operating Temperature</td>
<td>Heat sink temperature about 55°C</td>
</tr>
<tr>
<td></td>
<td>Shell temperature about 28~45°C</td>
</tr>
<tr>
<td>Water Protection</td>
<td>IP 62 (Dust tight and protection against dripping water)</td>
</tr>
</tbody>
</table>
Final Design
<table>
<thead>
<tr>
<th><strong>Final Design Specification</strong></th>
</tr>
</thead>
</table>

**107.95 mm**

**38.1 mm**

**LCD Display**

**Capacitive Touch**

**2 Microphone Array**

**Micro-USB Power**

<table>
<thead>
<tr>
<th><strong>Size</strong></th>
<th>1.5” x 4.25” x 4.25” (38mm x 108mm x 108mm)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Weight</strong></td>
<td>248.5 grams</td>
</tr>
<tr>
<td><strong>Screen</strong></td>
<td>1.3” (33 mm) Diameter Color TFT LCD Display</td>
</tr>
<tr>
<td><strong>Material</strong></td>
<td>PC Plastic</td>
</tr>
<tr>
<td><strong>Wi-Fi Connectivity</strong></td>
<td>Wi-Fi module providing fully integrated 2.4 GHz 802.11 b/g/n radio, TCP/IP stack and a 32-bit microcontroller (MCU)</td>
</tr>
<tr>
<td><strong>Audio</strong></td>
<td>Able to seamlessly connect and control your existing SONOS home network</td>
</tr>
<tr>
<td><strong>System Requirements</strong></td>
<td>COM, comes ready to connect to your Wi-Fi. Requires an iOS or Android device compatible with the SONOS app.</td>
</tr>
<tr>
<td><strong>Power</strong></td>
<td>5V Supply via wall wart adapter to micro USB</td>
</tr>
<tr>
<td><strong>Operating Temperature</strong></td>
<td>CPU temperature ~43°C Shell temperature 26 °C</td>
</tr>
<tr>
<td><strong>Water Protection</strong></td>
<td>IP 62 (Dust tight and protection against dripping water)</td>
</tr>
</tbody>
</table>
Microphone Placement Evolutions
Initial Placements
Next Considerations
Final Placement
Hardware Design
Functional Hardware Block Diagram
First Spin PCB
Final PCB
Comparison
MCU - NXP LPC4088

- ARM Cortex-M4 based digital signal controller.
- Features utilized
  - Three UARTs (Wifi/ISP)
  - I2S Rx (Mics)
  - I2C (Cap Touch)
  - SPI (LCD)
  - GPIO (ISP/RESET/IRQ)
- A general MCU that our instructors and TA are familiar with.
- Well Supported
- I2S Mics
- Ultimately, not the right MCU for this job. More on this in a later slide.
- Memory
  - 512 kB of flash program memory
  - up to 96 kB of SRAM data memory
  - up to 4032 byte of EEPROM data memory
WiFi - WF121 Module

- UFL connector for external antenna.
- Two UART connections only one with flow control
- Tx and Rx lines swapped due to labeling misunderstanding
- Supports 802.11 b/g/n
- RF shield
- Why this device?
  - Easy to use software library (bglib)
  - Supports WiFi b/g/n
  - Access Point mode and Standalone Client Mode
- Two UARTS
  - API
  - Data
- Available in two packages
Capacitive Touch Design

- Twelve possible input connections for capacitative control
- I2C communication
- Output IRQ signal for registering a touch
- Prevention of false triggering.
- Chose solution AD7142
  - 0 pf to 250 pf
  - 1.69 for 2,500
  - Twelve possible inputs
- 12.5 mm pads chosen as standard reflection and index finger size
Power System

- **Micro USB connector.**
  - Power cable is readily available in most homes.
  - PCB extension allows for ease of physical constraints
- **5V → 3.3V Voltage Regulator**
- **Output Voltage Ripple Tolerance of 1.5%**
- **-40°C to 125°C**
- **Low-dropout voltage 38 mV at 150mA load current**
Microphones - SPH0645LM4H-B

- I2S Output
  - Decimation is done directly in the microphone and eliminate the need for an ADC or codec
  - Fewer conversions
    Analog(voice) → Digital → Digital transmission → COM.
- Left and right Mic (Dual Channel)
- RF Shielded
- Omni-directional
- High SNR of 65dB(A)
- Frequency Response vs. Sensitivity
  (human voice: 85 Hz - 260Hz)
Microphones

- Perform under 3 different modes: active, sleep and powered off
- Align to the hole drilled to the outer case
- Control the data by word selecting signal and clock
Microphone Data

18 bits of resolution

8 bits of tristate

6 bits padded 0

Totalling 32 bits on each channel with a word select frequency 1/64 of the clock frequency.
Display - 2.2” Adafruit Display

- 2.2” display chosen to provide more screen real estate in final design.
- SPI interface
- Library ported from C++ to C and LPC Open framework
- Past experience with display
Inverted F Antenna

- Orientation, current location, and antenna choice is due to distancing the antenna away from the noise generated by the other components in our device.
- We used the Heatsink to our advantage as it shields the antenna from the rest of the components in our device. And it helps radiate the signals coming out from the antenna.

Precautions:

- The surfaces you place the COM. on will need to be taken into consideration. (i.e. Placing the devices on a metal surface will yield worse results than putting on a wooden surface.)
Software Design
Program runs directly on hardware.

C Language
No operating system.
LPC Open Framework.
Modular by design.

captouch/
mics/
screen/
util/
wlan/
main()
main()

// abridged version
int main(void) {
    master_init();
    delay_init();
    screen_init();
    mics_init();
    wlan_init(wlan_init_cb);
    captouch_init(captouch_handler);

    while(1) {
        wlan_process();
        mics_process();
        captouch_process();
    }

    return 0;
}
When you initialize this module, you provide a pointer to a function to handle touch events.

Simple to use interface.
• Samples are read as 32 bit integers via I2S. We retain 16 of the 18 valid bits of data.
  • Necessary due to memory constraints. And we can store in increments of 8 bits.
• Peripheral to Memory DMA is used to record audio into two buffers of equal size (explanation for why two buffers in next slide).
• Currently capturing 4 32-bit samples at a time into a uint32_t array, then, once DMA transfer has completed, moving into a uint16_t array for storage, only retaining 16 most significant bits.
mics/

Why is the recording split into two buffers?

• Not enough memory in the first RAM bank. So we’re using RAM and RAM2 banks on the 4088.
• I explicitly store half of the audio recording in another bank to leverage its additional storage.

```c
volatile uint8_t recording[RECORDING_SIZE];

__DATA(RAM2) volatile uint8_t recording2[RECORDING_SIZE];
```

• Actual recording size is 2 * RECORDING_SIZE.
• #define RECORDING_SIZE 24000 (48,000 bytes total)
Recording Data Flow

Audio Capture

Microphone I2S Input → DMA copies samples to buffer. → ISR moves buffer into larger storage array.

Loop until entire recording captured.

Transport recording to the player.
Recording Data Flow

Both methods are supported in code.
#define PUBLISH_DONT_SERVE
Recording Playback

• Could not use Method #1 (Built-in HTTP Server)
  
  Slower transfer speed from our device  
  +  
  player attempting to play before finished downloading  
  =  
  < 1 sec of audio played from 3 sec recording

• Method #2 (Proxy Server) as a solution for now.
wlan/

- WiFi code was designed as a state machine so that it will not block the CPU while waiting for responses and events from the WiFi module.
- UPnP
  - Handles basic UPnP commands to control SONOS devices.
  - Can issue requests to fetch device and track information.
  - Can discover devices (and rooms) on your home network.
- WAV audio file server
  - Provides an HTTP server that serves WAV file of recording.
- Generic HTTP request support.
  - Issues a custom HTTP request, and stores the response for further processing and analysis.
- Setup TCP server, used in the device setup process.
util/

- Delay Functions
  - Utilizes the system timers
  - Extremely accurate delay function
- Queue utility functions modified from third party.
screen/

- Ported over an existing C++ Adafruit library to C and our LPCOpen platform.
- Icons
  - Bitmaps indicate which pixels need to be colored
  - X BitMap
- Text
  - There are functions that take in a string and calculate which pixels need to be colored
Summary
void mics_init(void) {
  // mics_pin_mux_debug_pincheck(); // IF WIRES CONNECTED
  mics_pin_mux();
  I2S_AUDIO_FORMAT_T audio_Config;
  audio_Config.SampleRate = MIC_SAMPLE_RATE;
  /* Select audio data is 2 channels (1 is mono, 2
     audio_Config.ChannelNumber = 1;
  /* Select audio data is 16 bits */
     audio_Config.WordWidth = 32;

  Chip_I2S_Init(LPC_I2S);
  Chip_I2S_RxConfig(LPC_I2S, &audio_Config);
  Chip_I2S_RxModeConfig(LPC_I2S, 0, 0, 0);

  Chip_I2S_DMA_RxCond(LPC_I2S, I2S_DMA_REQUEST_CHAN
  /* Initialize GPDMA controller */
  Chip_GPDMA_Init(LPC_GPDMA);
  /* Setting GPDMA interrupt */
  NVIC_Enable_IRQ_DMAChannels(0x00, 0x00000000, 0x00000000);
}
Tangible Outcomes

Hardware

Physical Units & Test Results

Microphone and DSP Research

Embedded Software
It’s been designed and built.
void mics_init(void) {
    // mics_pin_mux_debug_pincheck(); //IF WIRES CONNECTED TO MICS NO NEED TO GO THROUGH DEBUG
    mics_pin_mux();
    I2S_AUDIO_FORMAT_T audio_cfg;
    audioCfg.SampleRate = MIC_SAMPLE_RATE; /* Mono, 2 is stereo */

    dmaChannelNum_I2S_Rx = Chip_GPDMA_GetFreeChannel(LPC_GPDMA, GPDMA_CONN_I2S_Channel_1);
    activeCb = NULL;

    NVIC_enableIRQ(DMA_IRQn);
}

It’s real.
void mics_init(void) {
    mics_pin_mux_debug_pincheck(); //IF WIRES CONNECTED TO MICS NO NEED TO GO THROUGH DEBUG
    mics_pin_mux();
    I2S_AUDIO_FORMAT_T audio_Config;
    audio_Config.SampleRate = MIC_SAMPLE_RATE;
    // is mono, 2 is stereo */
    NVIC_SetPriority(DMA_IRQn, ((0x01 << 3) | 0x01));
    NVIC_EnableIRQ(DMA_IRQn);
    dmaChannelNum_I2S_Rx = Chip_GPDMA_GetFreeChannel(LPC_GPDMA, GPDMA_CONN_I2S_Channel_1);
    activeCb = NULL;
}

It’s the COM.
Thank You
Sonos, Laritech, and UCSB

Any Questions?