Project Proposal Form

Team letter:	C	Name of person elected as team leader:	Nathan Ruttley

Responsibilities

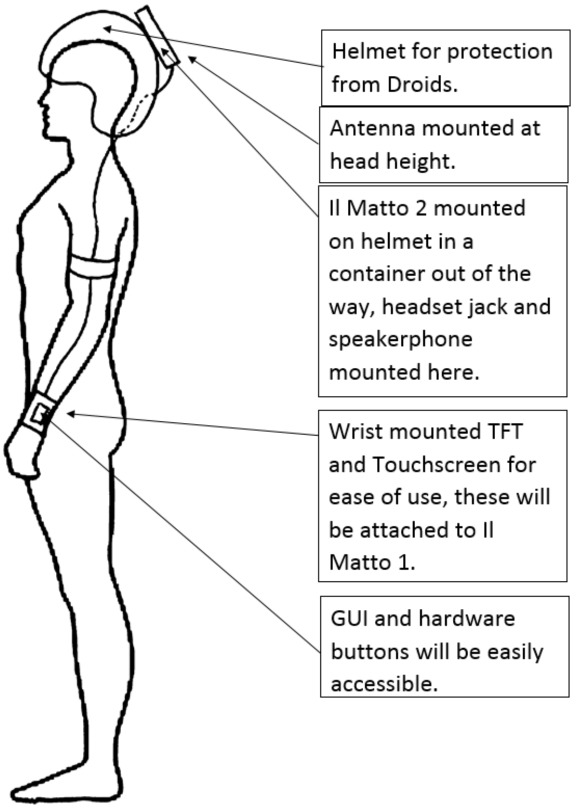
Lab pair no.	Name	Design responsibility
19	Fiona Moore	• Write/test receive function • Microphone interface • UART • Data types • Wireless transceiver code
15	Joe Sturgeon	• Edit configuration for wireless module library • Write/test transmit function • Speaker/ headphone interface • En/Decryption • Antenna
30	Diwen Hu	• Touchscreen interface design • UART communications protocol • Power management • Construction of wireless interface
29	Yubo Zhi	• Design of GUI, touchscreen and display implementation • UART communications protocol
27	Alaa Khoja	• Design, simulation and construction of microphone amplifier • Development of code to sample microphone • Development of code to store audio on SD card
7	Nathan Ruttley	• Design, simulation and construction of audio amplifier • Construction of Il Mattos • Development of code to output audio via PWM • Development of code to store audio on SD card • Team management

Overall Design Summary

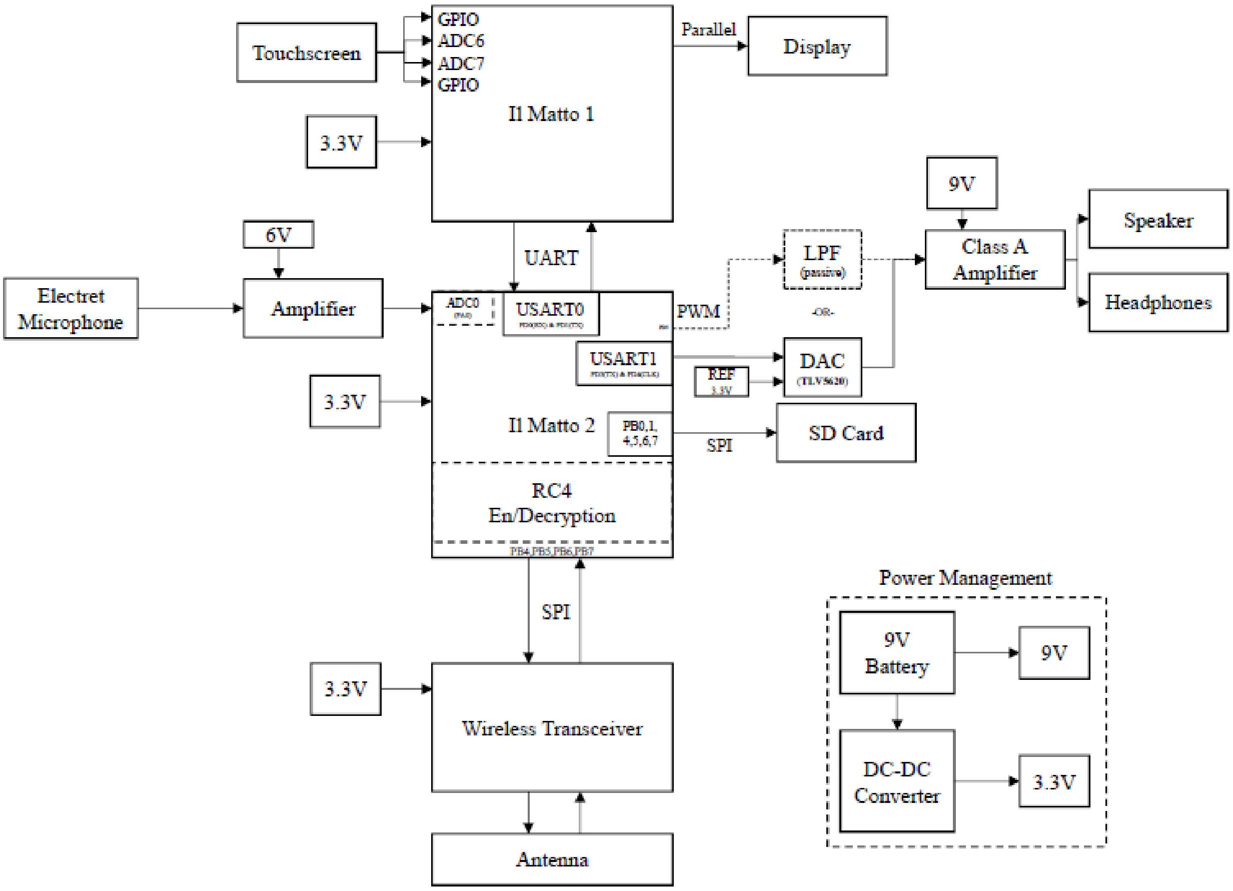
Complete solution will be a device capable of:

- Capturing audio over a frequency range of at least 85Hz to 255Hz from a microphone integrated into a headset or a built in microphone on the device
- Outputting audio in a headset or by loudspeaker
- Capturing written input from a resistive touchscreen
- Displaying written data on the screen of the device
- Authenticating a user
- Encrypting and decrypting data by RC4
- Transmitting and receiving data with a packet structure over a wireless link at a length of tens of meters.

The final packaging of the device will be as follows:


Helmet for protection from Droids.
Antenna mounted at head height.
Il Matto 2 mounted on helmet in a container out of the way, headset jack and speakerphone mounted here.
Wrist mounted TFT and Touchscreen for ease of use, these will be attached to Il Matto 1.
GUI and hardware buttons will be easily accessible.

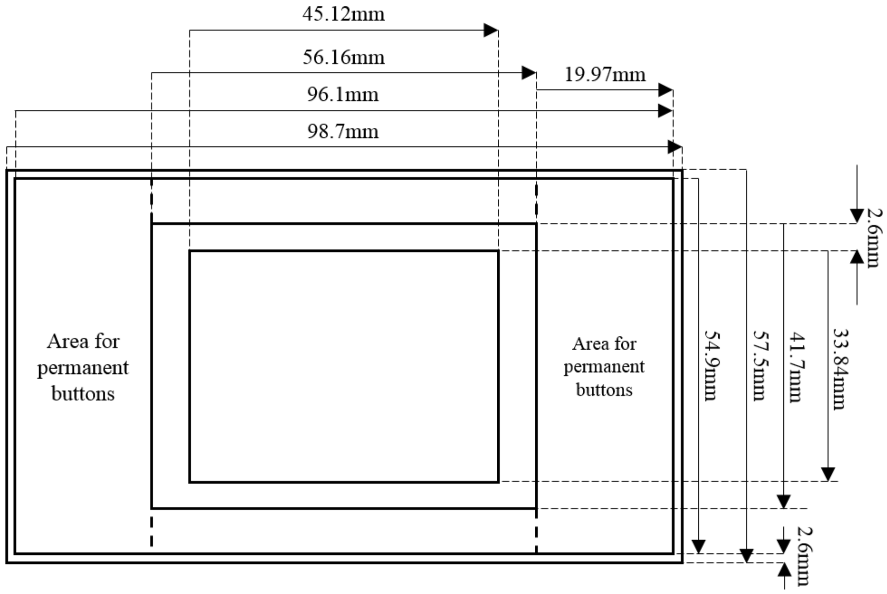
The specification of the device will be based upon the following top-level design (data sources to the left, data sinks to the right excluding SD Card), pin assignments have been included where necessary to avoid conflicts:



Subsections and description:

- Touchscreen
  - 4 line resistive touchscreen.
  - 98.7 x 57.5mm
  - Will need 2 ADCs to capture data and 2 GPIO
    - ADC6 and ADC7 on Il Matto 1
    - GPIO Pins to be confirmed depending on cable position of touchscreen
- Display
  - 2.2" TFT Display Module (Steve Gunn)
  - 8-bit parallel interface
    - Ports A and C on Il Matto 1, some unused pins on PORTA for touchscreen

The above two components will be assembled as follows:

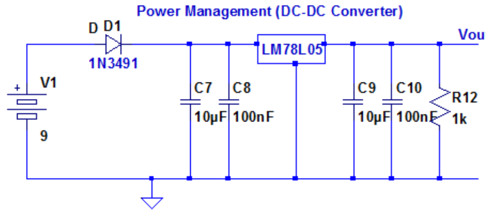


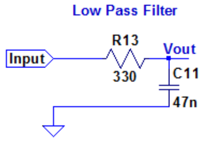
all mounted on Il Matto 1 on the arm of the user.

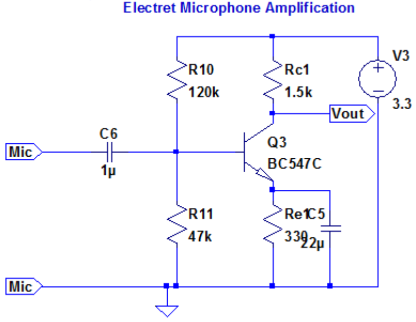
- Il Matto 1
  - Processes data from 2 ADC channels which are capturing data from the touchscreen
  - Pushes this data to Il Matto 2 when requested
  - Accepts data from Il Matto 2 to be pushed to the display
  - Tell Il Matto 2 when to start and stop sound transmission
  - Tell Il Matto 2 when to enter and leave power saving mode
  - Communicates via UART to Il Matto 2
- Microphone
  - Capable of capturing audio in the range of human speech
    - 300 Hz to 3400 Hz
    - *Although the range of human hearing is 20-20kHz, not amplifying frequencies other than those of human speech will equate to some background noise cancellation.*
  - Standard operating voltage of 2V.
  - Maximum current consumption of 0.5mA.
  - Sensitivity of -42±3 dB
  - Disabled upon connection of a headset
- Microphone amplifier
  - Amplifies voltages of approximately 20mV to 3.3V from microphone to voltages of 0 to 3.3 for connection to ADC
  - Gain of around 165, in the case the output of microphone is 20mV.
  - has a 3dB bandwidth of around 4MHz.
- Il Matto 2
  - Captures data from microphone
  - Converts amplified microphone data into digital format using ADC unit.
  - Stores and fetches data from SD card

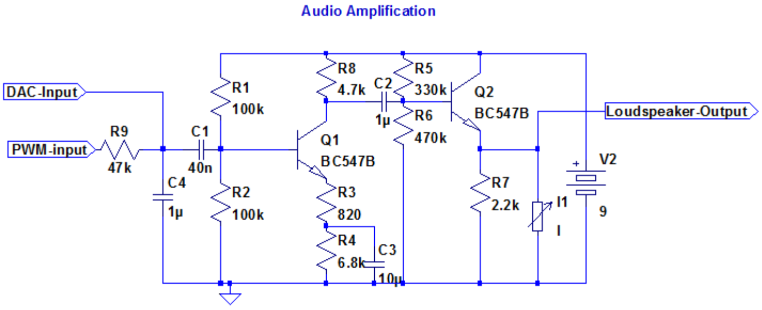
    - Performs RC4 encryption and decryption on data
    - Interfaces with wireless transceiver
    - Outputs audio by pulse width modulation to low pass filter
    - Interfaces with Il Matto 1 to send visual data or receive touchscreen data
- Low pass filter
    - Creates analogue voltage from PWM output of Il Matto 2
- DAC - TLV5620
    - 8-Bit DAC
    - Communicates with Il Matto 2 via USART1 in SPI mode
        - Pins PD4,3
- SD Card
    - Stores data
- Audio Amplifier
    - Amplifies voltages of 0 to 3.3 from LPF to voltages of -8 to 8 for loudspeaker
    - Has a gain of 4.8
    - Has a 3dB bandwidth of at least 170Hz, centred around 170Hz (85 to 255Hz for human voice)
    - Class A
- Speaker
    - Required input voltage of -8 to 8
    - Will have SPL least 86dB
- Wireless Transceiver
    - RFM12B-S2 wireless transceiver
    - Communicates with Il Matto 2 over SPI in packets of data which have length and type
    - Requires voltage of 3.3V (can be powered by Il Matto 2)
    - Requires PB2,4,5,6,7 on Il Matto 2
- Antenna
    - Length of 16.5cm
- Encryption and Decryption
    - RC4 algorithm so encryption and decryption are the same operation
- Power management
    - Will utilize a DC-to-DC converter as opposed to a monolithic voltage regulator to minimize power wastage

Initial circuit diagrams have been drafted for power management, filters and amplifiers, these are shown below:









After testing with real components these circuits may need to be adjusted. In particular the gain of the microphone amplifier and the input to the audio amplifier (no LPF needed if a DAC is used as opposed to PWM to output audio)

Module Design Proposals

Names of people involved:	Fiona Moore, Joseph Sturgeon, Diwen Hu
Title of Module:	Data transmit and receive
Module Details:	Transmitting and receiving data over wireless RFM12B module Interfacing wireless module with Il Matto 2 over SPI ○ Communicates with Il Matto 2 over SPI in packets of data which have length and type ○ Requires voltage of 3.3V (can be powered by Il Matto 2) ○ Requires PB2,4,5,6,7 on Il Matto 2 UART interface with Il Matto 1 En/decryption (RC4) ○ RC4 algorithm so encryption and decryption are the same operation

Names of people involved:	Yubo Zhi, Diwen Hu
Title of Module:	Tactile input and visual output
Module Details:	Interfacing with 4 line resistive touchscreen Interfacing with 2.2” TFT display UART interface with Il Matto 2 Creation of user interface

Names of people involved:	Alaa Khoja, Nathan Ruttley
Title of Module:	Audio input and output
Module Details:	Amplification of microphone input: • Capable of capturing audio in the range of human speech • 85Hz to 225Hz • Standard operating voltage of 2V. • Maximum current consumption of 0.5mA. • Sensitivity of -42±3dB. • On board microphone disabled upon connection of a headset • Microphone amplifier ○ Amplifies voltages of approximately 20mV to 3.3V from microphone to voltages of 0 to 3.3 for connection to ADC ○ Gain of around 165 , in the case the output of microphone is 20mV ○ Has a 3dB bandwidth of around 4MHz. ○ Circuit components consist of: 2 10k resistors, 100k, 2 100nF and 1 1uF capacitor and 1 BC547C transistor. Capture of data from microphone input • ADC in free running mode for maximum sampling

	• Immediate transmission of data over wireless or to SD card – else data is discarded Amplification of Il Matto audio output (PWM to audio or DAC) (Class A amp) Storage and retrieval of audio from Il Matto to SD card

Names of people involved:	Alaa Khoja, Nathan Ruttley, Diwen Hu
Title of Module:	Power management
Module Details:	DC-DC conversion of 9V to 3.3V for use with Il Mattos and 5V for use with amplifiers . 9V input use L78L05 voltage regulator to get 5V output. . 5V input use 1117LV33 voltage regulator to get 3.3V output in the same circuit. . Four capacitor are going to connect in parallel, two of them were 10uF and others were 0.1uF, and one more 1kΩ resistor. . This circuit can output two voltage at the same time, and it will adapt to the needs of different circuit.

Cost Estimates

Quantity (per device)	Item	Price (prototype)
1	4-way RCA Jack Connector	0.384
1	Electret Microphone	0.905
1	Headset	2.31
1	Loudspeaker	1.15
1	9V Battery Connector	0.522
2	Il Matto	20.00
1	Displays	10.00
3	Transistors	0.03
	Passive components	0.01
1	Stripboard	2.50
1	Wireless Transceiver	8.40
1	Voltage Regulator	4.95
1	Touchscreen	12.39
1	DAC	0.66
Quantity (per device)	Item	Price (large scale production)
1	4-way RCA Jack Connector	0.384
1	Electret Microphone	0.56
1	Headset	2.31
1	Loudspeaker	0.92
1	9V Battery Connector	0.27
2	Il Matto	20.00
1	Displays	10.00
3	Transistors	0.01
	Passive components	0.01
1	Stripboard	1.45
1	Wireless Transceiver	8.40
1	Voltage Regulator	0.32
1	Touchscreen	11.87
1	DAC	0.66
6	Construction hours @ £10/Hour	60
Fixed Costs QTY	Item	Price
100	Software Development @ £75/hr	7500
15	Person-hours (design and debugging) @ £75/hr	1125
1	Conformance Testing	2000
1	Overheads	100000

Total Cost Per Prototype (components and construction)	£123.211
Fixed Costs	£110625
Total Design cost	**£110751.422**
Large scale device cost	£118.15
Unit Sale Price (excluding VAT)	£185.18 (£67.03 profit per device)
Unit Sale Price (including VAT@20%)	£222.22
Required Sale Quantity (90% Yield) (average £60.33 profit)	1836 units (breakeven) <20% of remaining populous

If 50% of the remaining population bought our SPECIES we would turn a £190884.12 profit.

Prototyping and Construction Method

- Prototype circuits will first be constructed on breadboard.
- After successful testing and debugging the circuits will be constructed on strip-board and soldered.
    - These circuits will be tested again after soldering.
- Direct wires to inputs and outputs of components/Il Mattos will be used where possible, especially for extensions down arms (see final product appearance) so as to minimise the risk of connections breaking.
- The head-mounted Il Matto will be in a waterproof, non-shatter, plastic container along with the batteries.
- The wrist mounted Il Matto and screen will be housed in a plastic container, also watertight and attached via a fabric/elastic wristband
- All wire connections will be wrapped in a heat shrink material to ensure ruggedness.

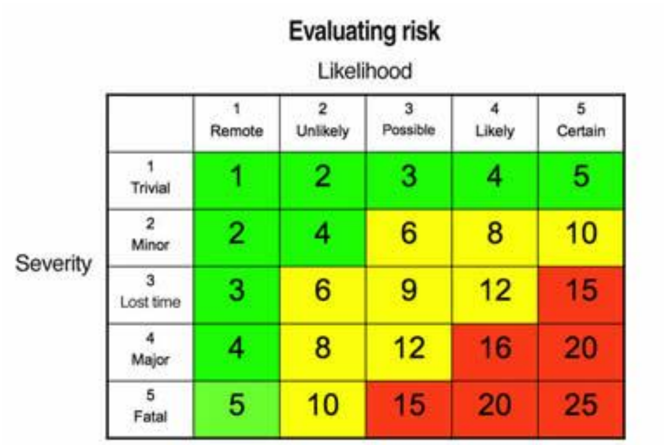
Surface mount packages will be avoided, all components will be purchased in their through-hole variants.

Planned Project Activities

Activity	Initials	Fri am	Fri pm	Mon am	Mon pm	Tue	Wed	Thu	Fri am	Fri pm	Mon am	Mon pm
Building, testing and debugging microphone amplifier, and confirm the gain value.	AK	✓										
Construction, test and adjustment of audio amplifier (on breadboard)	NER			✓								
Output audio data to DAC and test functionality - Adjust code if needed	NER				✓	✓						
Modifier amplifier circuit for use with headphone jack	NER					✓	✓					
test modified version of amplifier+ with microphone	AK			✓	✓	✓						
Build finalized design of amplifier and input data to ADC through microphone.	AK						✓	✓				
Write and test the receive function for wireless module	FM	✓										
Write and test the transmit function for wireless module	JMS	✓										
Integrate test and receive functions	FM		✓									
Integrate test and receive functions	JMS		✓									
Test transmitting and receiving a sine wave	FM			✓	✓							
En/decryption	JMS			✓	✓							
Communication between both Il Mattos (UART)	YZ					✓	✓	✓	✓	✓		
Build RC LPF	DH	✓										
Build and test DC-DC converter	DH					✓	✓					
Touch screen interface design	DH				✓	✓		✓	✓			
Touch screen interface design	YZ			✓	✓	✓	✓					

GUI design	YZ	✓	✓	✓	✓	✓	✓					
Touchscreen sketching	YZ				✓	✓	✓					
Touchscreen edge permanent buttons design	YZ					✓	✓	✓				
Test transmitting and receiving touchscreen sketch	YZ						✓	✓	✓	✓		
Test DAC chip output	YZ	✓	✓									
Soldering of final circuits	NER										✓	✓
Oversee integration of entire system	NER										✓	✓

Risk Management



Evaluating risk

Likelihood

Severity

	1 Remote	2 Unlikely	3 Possible	4 Likely	5 Certain
1 Trivial	1	2	3	4	5
2 Minor	2	4	6	8	10
3 Lost time	3	6	9	12	15
4 Major	4	8	12	16	20
5 Fatal	5	10	15	20	25

International Register of Certified Auditors (IRCA), “A History of Risk”, http://www.irca.org/Global/Images/technical/inform/issue%2024/24-SAsbury-Figure1.jpg

Hazard	Severity	Likelihood	Risk	Control	Controlled Severity	Controlled Likelihood	Controlled Risk
Components are damaged/broken through misuse	3	4	12	Comply with ESD handling guidelines. Confirm correct wiring with datasheet before applying power. Turn off power before rewiring. Order a spare of key components, if budget permits.	2	2	4
Illness of a team member	4	3	12	Early intervention, stay well rested, eat well, and make time to relax. Ensure that all team members are aware of what work others are doing so that they can continue work of an ill person.	3	2	6
Lack of contribution from a team member	4	2	8	Ensure everybody has a job to do and is capable of doing it. Ask for help rather than not doing anything. Encourage each other.	3	2	6
Lack of available time	3	4	12	Organise time well, divide up tasks to make best use of available time. Plan ahead.	2	4	8
Lack of access to labs	3	4	12	Ensure that files on computers can be accessed outside of labs. Do work that doesn’t require access to labs.	2	4	8
Components delayed/DOA	4	4	16	Start assembling components that are available. Research alternatives that are readily available.	2	4	8
Failure of key design modules/incompatibility	5	4	20	Test sub-systems before final assembly. Agree communications protocols between modules before building them. Use debugging knowledge of the team to attempt to fix module.	4	3	12

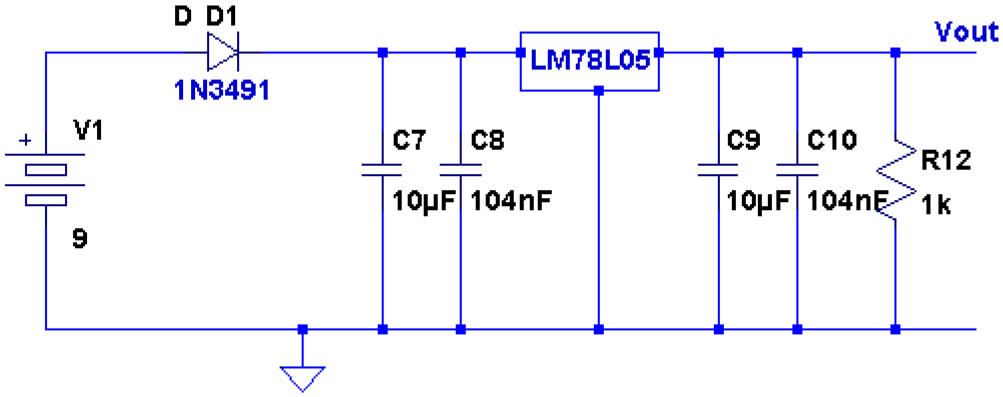
Initial Software Listings

Project Milestones

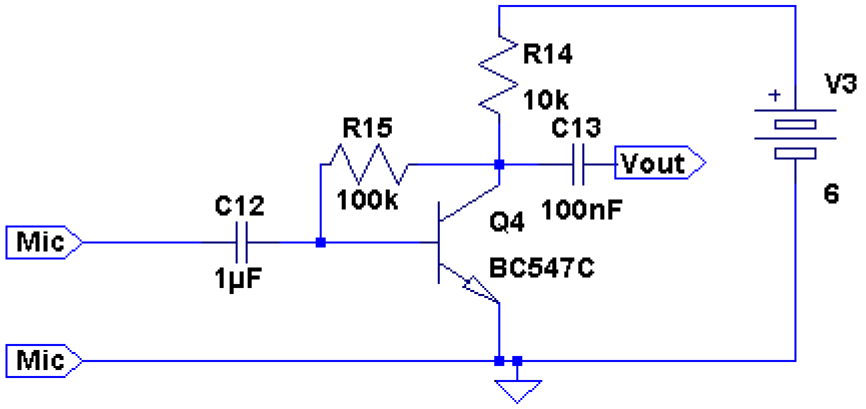
Component of system/Milestone	Supervisor	Time/Date	Comments (all/part/none working; protoboard/constructed)
Wireless modules interfaced with Il Mattos			
Speech sampled by Il Matto			
Wireless communication of speech between Il Mattos			
Bi-Directional Voice communication between devices			
En/Decryption of data			
Transmition of encrypted audio and successful decryption			
Il Matto input to audio amplifier can drive loudspeaker			
Touchscreen input captured and displayed on TFT display			
Touchscreen input displayed on other device			
GUI created for user authentication			
Voice memos stored and retrieved on SD card			
Power management of system			
Integration of complete system			

System implemented as a wearable device			

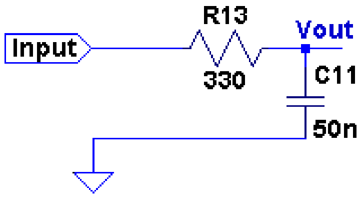
Power Management (DC-DC Converter)



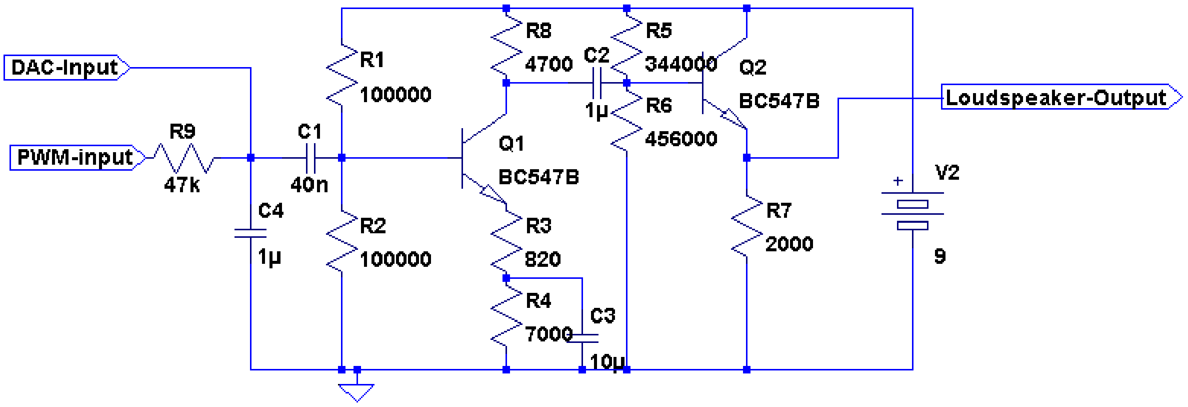
Electret Microphone Amplification



Low Pass Filter



Audio Amplification



--- H:\D4\amplifiers\ProjectProposalCircuitry.asc ---

```
#ifndef COMMUNICATION_H
#define COMMUNICATION_H

// UART baudrate (750,000 bps)
#define BAUD        (F_CPU / 8 / (1 + 1))

// Response
#define COM_ACK     0
// End of variable length response
#define COM_END     0xFF

// Data structure: Type, Data
// No data, response {COM_ACK} indicates IlMatto2 exist
#define COM_PING    0
// Wakeup wireless module, response {COM_ACK}
#define COM_WAKEUP  1
// Suspend wireless module for power saving, response {COM_ACK}
#define COM_SUSPEND 2

// Wireless connection operations
// Ping for other end, no data, response {COM_ACK} for success, {COM_END} for timeout
#define COM_W_PING  100
// Start sending & receiving sound data, no data, response {COM_ACK}
#define COM_W_SOUND 101
// Stop sending & receiving sound data, no data, response {COM_ACK}
#define COM_W_SOUND_END 102
// Send data to other end, data {Length(2 bytes), Data}, response {COM_ACK}
#define COM_W_SEND  103

#endif
```

```
#ifndef UART0_H
#define UART0_H

#include <stdio.h>

namespace uart0
{
    FILE *init(void);
    int getchNonBlocking(void);
    char getch(void);
    void putch(const char c);
    void poolSending(void);
}

#endif
```

```
#include <avr/io.h>
#include <stdio.h>
#include "uart0.h"
#include "communication.h"

void uart0::poolSending(void)
{
    while (!(UCSR0A & _BV(UDRE0)));
}

void uart0::putch(const char c)
{
    //if (ch == '\n')
    //  putchar('\r');
    poolSending();
    UDR0 = c;
}

static int putch(char ch, FILE *stream)
{
    uart0::putch(ch);
    return ch;
}

int uart0::getchNonBlocking(void)
{
    if (!(UCSR0A & (1<<RXC0)))
        return -1;
    return UDR0;
}

char uart0::getch(void)
{
    while (!(UCSR0A & (1<<RXC0)));
    return UDR0;
}

static int getch(FILE *stream)
{
    return uart0::getch();
}

FILE *uart0::init(void)
{
    #include <util/setbaud.h>
    DDRD &= ~0x03;
    PORTD |= 0x03;
    UBRR0H = UBRRH_VALUE;
    UBRR0L = UBRRL_VALUE;
    UCSR0A = USE_2X << U2X0;
    UCSR0B = (1 << RXEN0) | (1 << TXEN0);
    UCSR0C = (1 << UCSZ00) | (1 << UCSZ01);
    return fdevopen(::putch, ::getch);
}
```

```
#include <avr/io.h>
#include <stdio.h>
#include <util/delay.h>
#include <tft.h>
#include <portraitlist.h>
#include <landscapelist.h>
#include "menu.h"

#define LANDSCAPE

tft_t tft;
#ifdef LANDSCAPE
LandscapeList l(&tft);
#else
PortraitList l(&tft);
#endif

void init(void)
{
    DDRB |= 0x80;           // LED
    PORTB |= 0x80;
    tft.init();
#ifdef LANDSCAPE
    tft.setOrient(tft.FlipLandscape);
#else
    tft.setOrient(tft.Portrait);
#endif
    tft.setBackground(0x0000);
    tft.setForeground(0x667F);
    tft.clean();
    stdout = tftout(&tft);
    tft.setBGLight(true);
}

int main(void)
{
    init();

    tft.clean();
    tft.setForeground(0x0000);

    l.refresh();
    l.display(&menuRoot);

#if 0
#ifdef LANDSCAPE
    tft.rectangle(tft.topEdge() - 1, 0, 1, tft.height(), 0xF800);
    tft.rectangle(tft.bottomEdge(), 0, 1, tft.height(), 0xF800);
#else
    tft.rectangle(0, tft.topEdge() - 1, tft.width(), 1, 0xF800);
    tft.rectangle(0, tft.bottomEdge(), tft.width(), 1, 0xF800);
#endif
#endif

    uint16_t max = l.maxScroll(), v = 0;
inc:
    l.setScroll(v);
    _delay_ms(10);
    if (++v < max)
        goto inc;
    _delay_ms(1000);
dec:
    l.setScroll(v);
    _delay_ms(10);
    if (--v > 0)
        goto dec;
    _delay_ms(1000);
    goto inc;

    return 1;
}
```

```
#ifndef MENU_H
#define MENU_H

#include <list.h>

extern listItem menuRoot;

#endif
```

```
#include "menu.h"
#include <list.h>

static listItem item[40] = {
    // name, items, parent, func
    {"Item 1", 0, 0, 0},
    {"Item 2", 0, 0, 0},
    {"Item 3", 0, 0, 0},
    {"Item 4", 0, 0, 0},
    {"Item 5", 0, 0, 0},
    {"Item 6", 0, 0, 0},
    {"Item 7", 0, 0, 0},
    {"Item 8", 0, 0, 0},
    {"Item 9", 0, 0, 0},
    {"Item 10", 0, 0, 0},
    {"Item 11", 0, 0, 0},
    {"Item 12", 0, 0, 0},
    {"Item 13", 0, 0, 0},
    {"Item 14", 0, 0, 0},
    {"Item 15", 0, 0, 0},
    {"Item 16", 0, 0, 0},
    {"Item 17", 0, 0, 0},
    {"Item 18", 0, 0, 0},
    {"Item 19", 0, 0, 0},
    {"Item 20", 0, 0, 0},
    {"Item 21", 0, 0, 0},
    {"Item 22", 0, 0, 0},
    {"Item 23", 0, 0, 0},
    {"Item 24", 0, 0, 0},
    {"Item 25", 0, 0, 0},
    {"Item 26", 0, 0, 0},
    {"Item 27", 0, 0, 0},
    {"Item 28", 0, 0, 0},
    {"Item 29", 0, 0, 0},
    {"Item 30", 0, 0, 0},
    {"Item 31", 0, 0, 0},
    {"Item 32", 0, 0, 0},
    {"Item 33", 0, 0, 0},
    {"Item 34", 0, 0, 0},
    {"Item 35", 0, 0, 0},
    {"Item 36", 0, 0, 0},
    {"Item 37", 0, 0, 0},
    {"Item 38", 0, 0, 0},
    {"Item 39", 0, 0, 0},
    {"Item 40", 0, 0, 0},
};

static const listItem *rootItems[] = {
    &item[0], &item[1], &item[2], &item[3],
    &item[4], &item[5], &item[6], &item[7],
    &item[8], &item[9], &item[10], &item[11],
    &item[12], &item[13], &item[14], &item[15],
    &item[16], &item[17], &item[18], &item[19],
    &item[20], &item[21], &item[22], &item[23],
    &item[24], &item[25], &item[26], &item[27],
    &item[28], &item[29], &item[30], &item[31],
    &item[32], &item[33], &item[34], &item[35],
    &item[36], &item[37], &item[38], &item[39],
    0};

// name, items, parent, func
listItem menuRoot = {"Root", rootItems, 0, 0};
```