

Wake-Up-Word Speech Recognition in FPGA

Adolf A Dcosta • ECE 5570 • Dec 1st 2019

What's a FPGA ?

FPGA stands for field programmable gate array. Is an integrated circuit designed to be configured by a customer or a designer after manufacturing – hence the term "field-programmable". The FPGA configuration is generally specified using a hardware description language (HDL), similar to that used for an Application-Specific Integrated Circuit (ASIC). ^[1]

FPGAs contain an array of programmable logic blocks, and a hierarchy of "reconfigurable interconnects" that allow the blocks to be "wired together", like many logic gates that can be inter-wired in different configurations. Logic blocks can be configured to perform complex combinational functions, or merely simple logic gates like AND and XOR. ^[1]

What's a “Wake Up Word” ?

A wake word is a special word or phrase that is meant to activate a given device via voice. It is also referred to as hot word, trigger word, and wake up word. ^[2] A wake word needs to be unique. ^[2]

The objective of Automatic Speech Recognition (ASR) is to address the issue of creating a system that maps an acoustic signal into a string of words. ^[3] Speech as a computer interface has many benefits over conventional mouse and keyboard interfaces: speech is natural for humans, no special training is required, using speech multitasking is improved by leaving the hands and eyes free, and is often faster and more efficient to transmit than the information provided using conventional input methods. ^[3]

This kind of system is continuously listening and monitoring acoustic inputs which removes the necessity of non speech activation. ^[3]

Dev Kit DE2i-150

[User Manual Link](#)

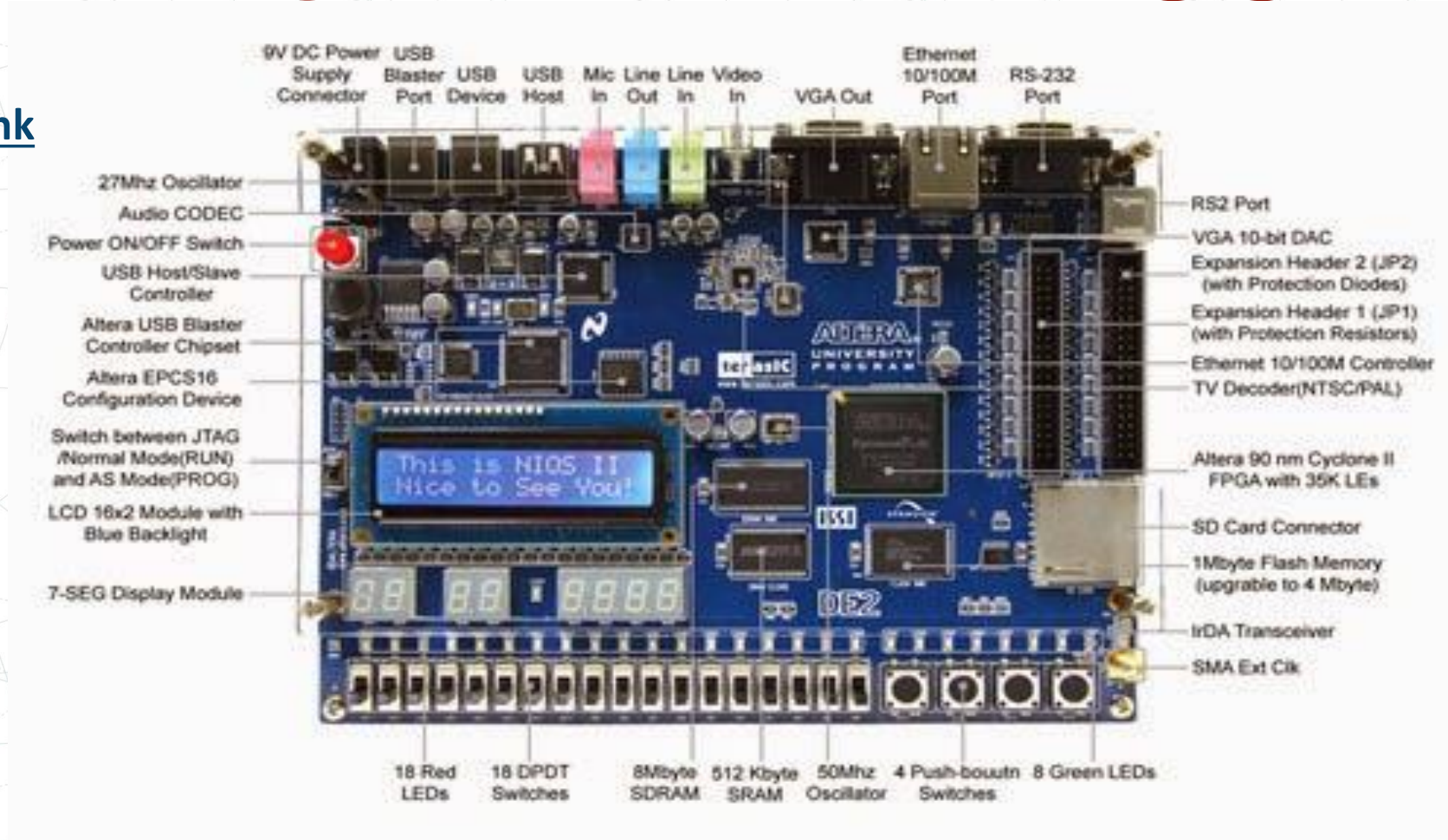


Fig1: DE2i-150 Layout Diagram^[4]

DE2i-150 Block Diagram

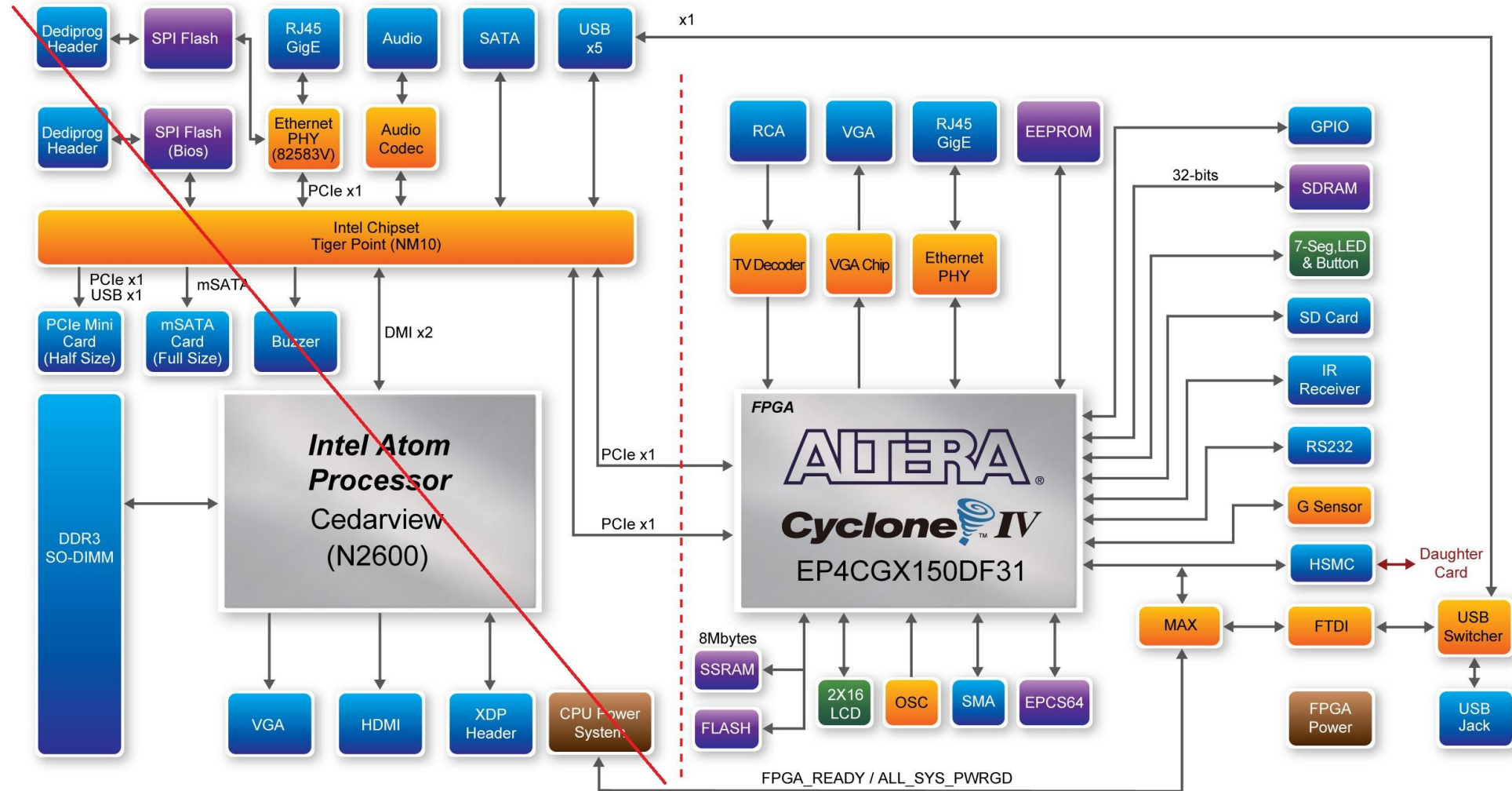
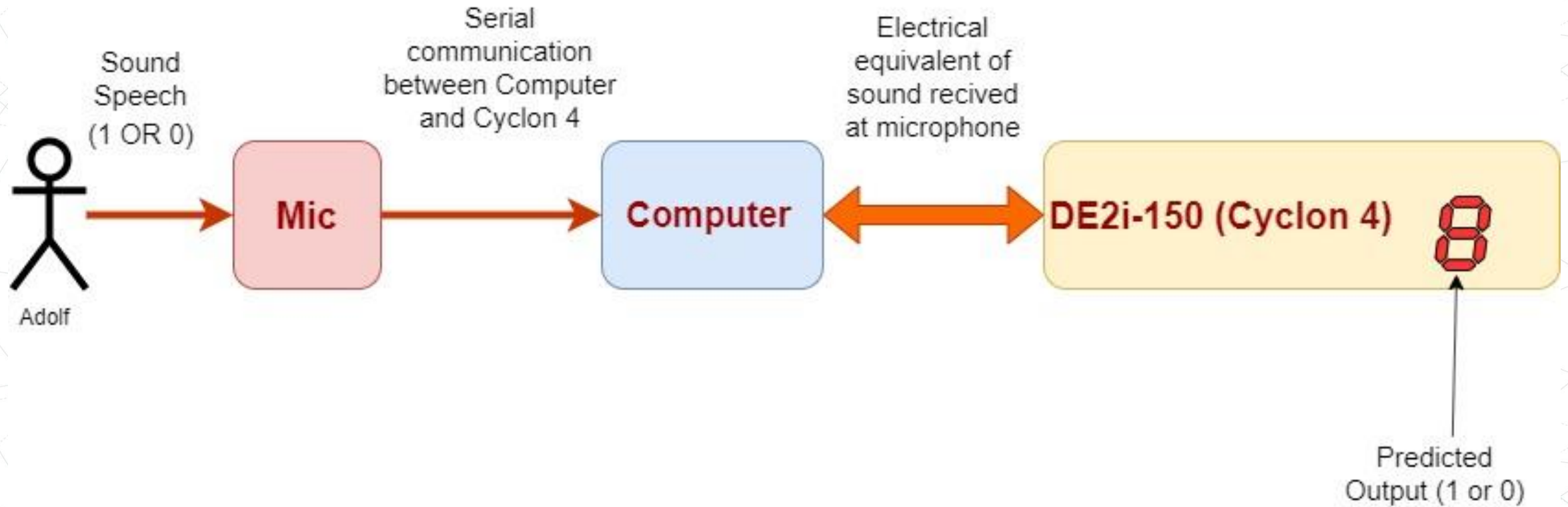


Fig2: DE2i-150 Block Diagram^[5]

DE2i-150 (FPGA Hardware)

- Altera Cyclone® IV 4CX150 FPGA device
- Altera Serial Configuration device – EPCS64.
- USB Blaster (on board) for programming; both JTAG and Active Serial (AS) programming modes are supported.
- Two 2MB SSRAM, Two 64MB SDRAM.
- 64MB Flash memory.
- SD Card socket.
- 4 Push-buttons.
- 18 Slide switches.
- 18 Red user LEDs, 9 Green user LEDs.
- 50MHz oscillator for clock sources.
- VGA DAC (8-bit high-speed triple DACs) with VGA-out connector.
- TV Decoder (NTSC/PAL/SECAM) and TV-in connector.
- Gigabit Ethernet PHY with RJ45 connectors.
- RS-232 transceiver and 9-pin connector.
- IR Receiver

Operational Block Diagram



Operation

The performance and the capability of the system are divided into two parts the front and the back end. The front end was running on a computer that runs a MATLAB code. The code is made simple with a GUI on a click of a button, the code is designed to take a sample of 1 second and store it in a double-precision array. Fourier Transform and windowing is applied, after which the graph is plotted on MATLAB in Frequency vs. Amplitude domain. The Synthesized data is then transmitted to the FPGA via the serial port using the USB to Serial converter. ^[6]

Serial Communication Configurations:

Criteria	Value
Bits Per Second	9600
Parity	None
Stop Bit	1
Flow Control	Hardware

Operation

There are only 1000 samples sent from the entire data to the FPGA for further processing through the USB to TTL which is the serial port. The data received is compared with the saved vectors.^[6] From the training file the Euclidean distance is compared, and the weight is calculated. The right one is given bigger weights and the result is displayed on the seven-segment display. The back-end system runs on 4 steps Receiving, Calculating Distance, Decision Making & Displaying Results.

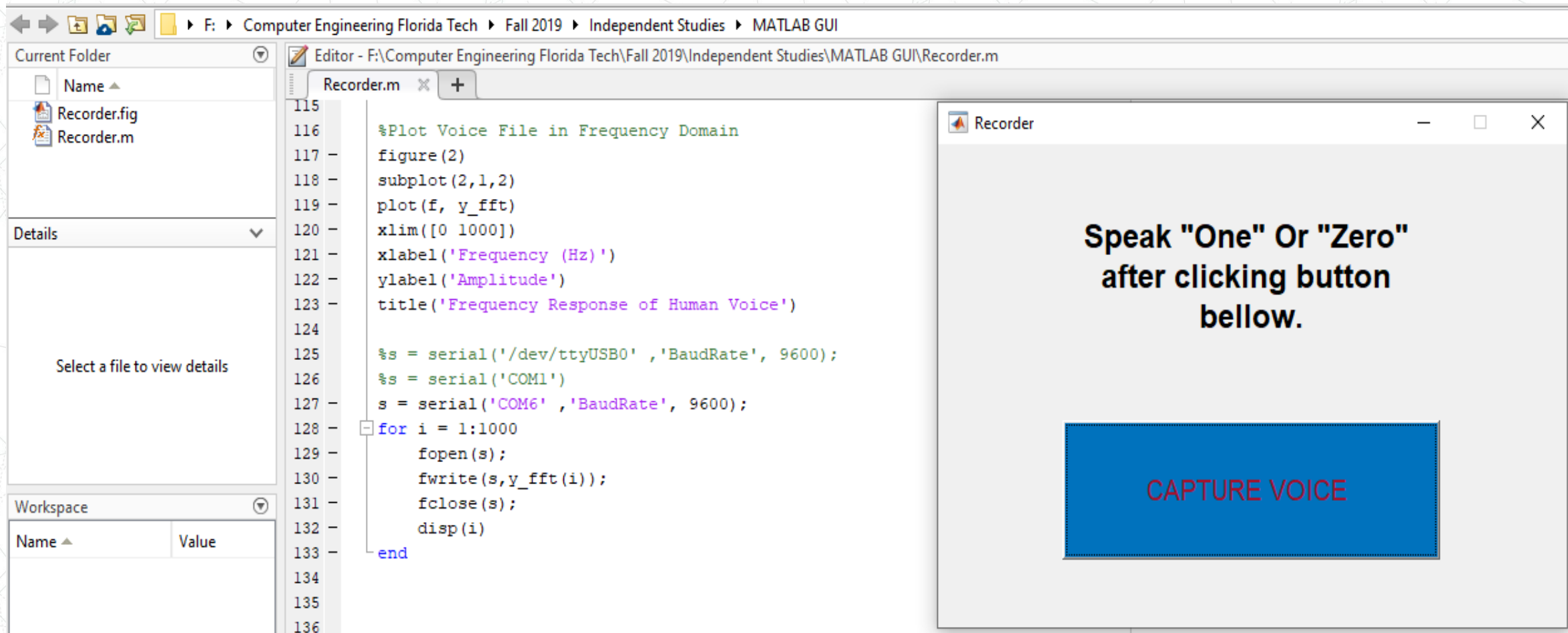
Programs Running on FPGA :

File name	Hierarchy
Voice_Recognition	Top Level Entity
Uart	
Uart_parity	
Uart_rx	
Uart_tx	

Programs Running on MATLAB :

File name
Recorder.m
Recorder.fig
Recorder.asv

GUI MATLAB



The image displays the MATLAB GUI Recorder interface. The top window shows the file explorer and the code editor. The code editor contains the following MATLAB code:

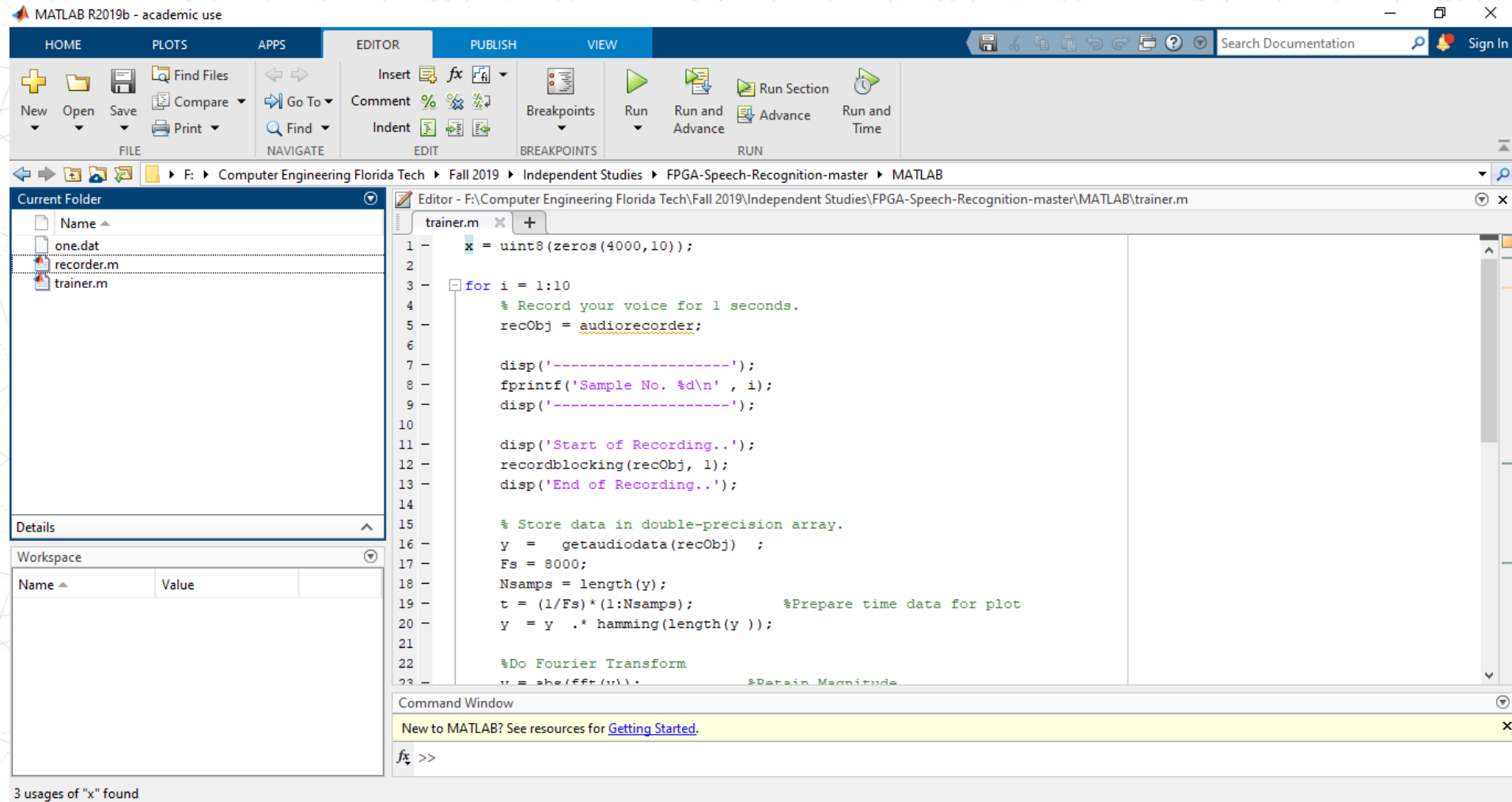
```
115  
116 %Plot Voice File in Frequency Domain  
117 figure(2)  
118 subplot(2,1,2)  
119 plot(f, y_fft)  
120 xlim([0 1000])  
121 xlabel('Frequency (Hz)')  
122 ylabel('Amplitude')  
123 title('Frequency Response of Human Voice')  
124  
125 %s = serial('/dev/ttyUSB0', 'BaudRate', 9600);  
126 %s = serial('COM1')  
127 s = serial('COM6', 'BaudRate', 9600);  
128 for i = 1:1000  
129     fopen(s);  
130     fwrite(s, y_fft(i));  
131     fclose(s);  
132     disp(i)  
133 end  
134  
135  
136
```

The bottom window, titled "Recorder", contains the following text:

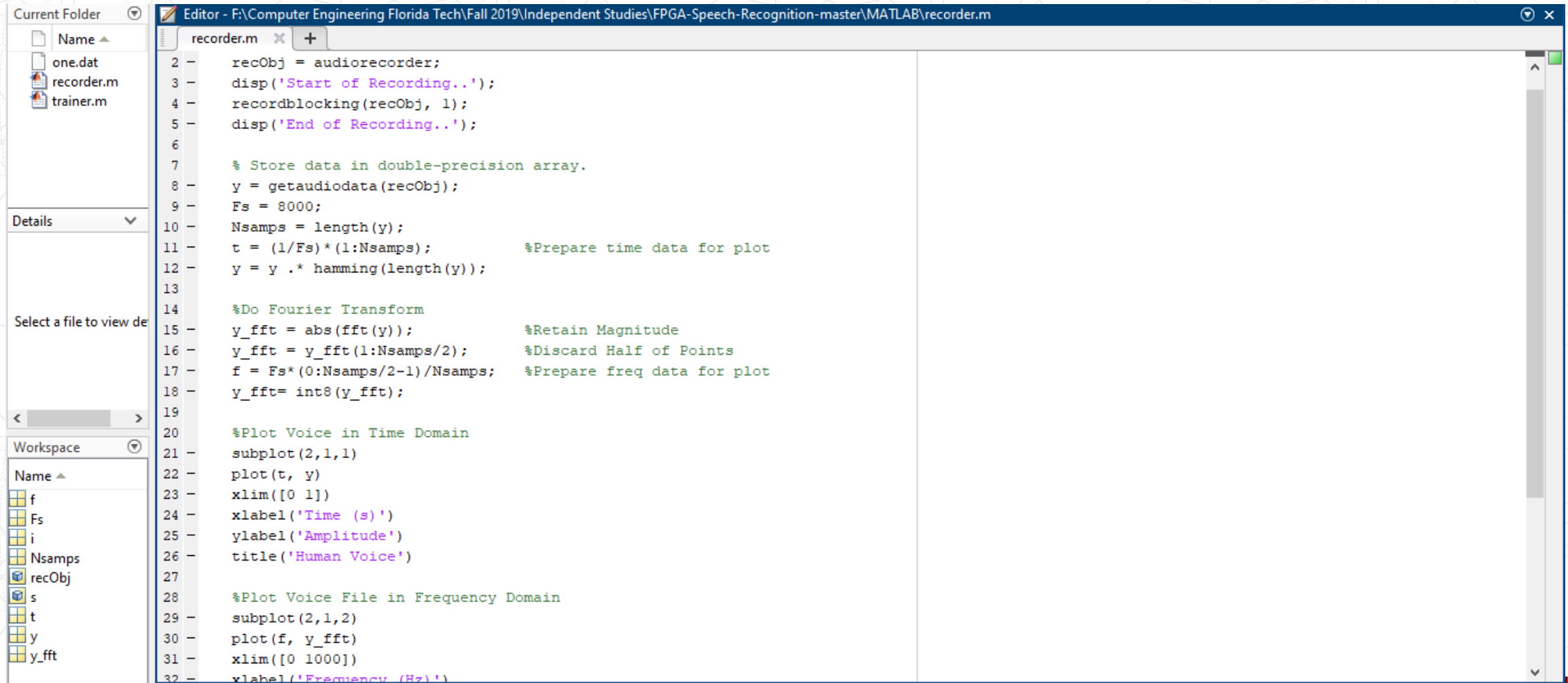
Speak "One" Or "Zero"
after clicking button
bellow.

CAPTURE VOICE

MATLAB



Record and TX data MATLAB



```
1 recorder.m
2 recObj = audiorecorder;
3 disp('Start of Recording..');
4 recordblocking(recObj, 1);
5 disp('End of Recording..');
6
7 % Store data in double-precision array.
8 y = getaudiodata(recObj);
9 Fs = 8000;
10 Nsamps = length(y);
11 t = (1/Fs)*(1:Nsamps); %Prepare time data for plot
12 y = y .* hamming(length(y));
13
14 %Do Fourier Transform
15 y_fft = abs(fft(y)); %Retain Magnitude
16 y_fft = y_fft(1:Nsamps/2); %Discard Half of Points
17 f = Fs*(0:Nsamps/2-1)/Nsamps; %Prepare freq data for plot
18 y_fft = int8(y_fft);
19
20 %Plot Voice in Time Domain
21 subplot(2,1,1)
22 plot(t, y)
23 xlim([0 1])
24 xlabel('Time (s)')
25 ylabel('Amplitude')
26 title('Human Voice')
27
28 %Plot Voice File in Frequency Domain
29 subplot(2,1,2)
30 plot(f, y_fft)
31 xlim([0 1000])
32 xlabel('Frequency (Hz)')
```

Current Folder

- one.dat
- recorder.m
- trainer.m

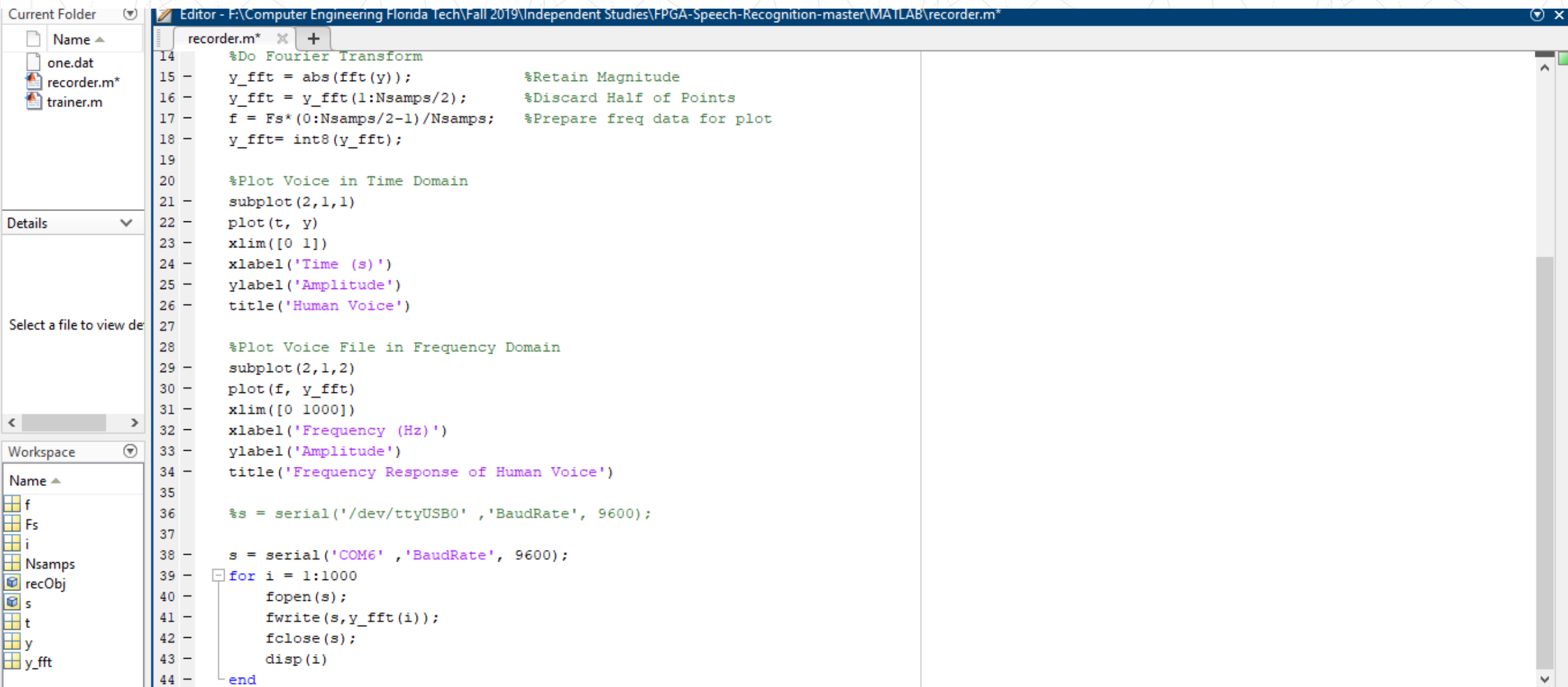
Details

Select a file to view details

Workspace

- f
- Fs
- i
- Nsamps
- recObj
- s
- t
- y
- y_fft

Record and TX data MATLAB



Current Folder

- one.dat
- recorder.m*
- trainer.m

Details

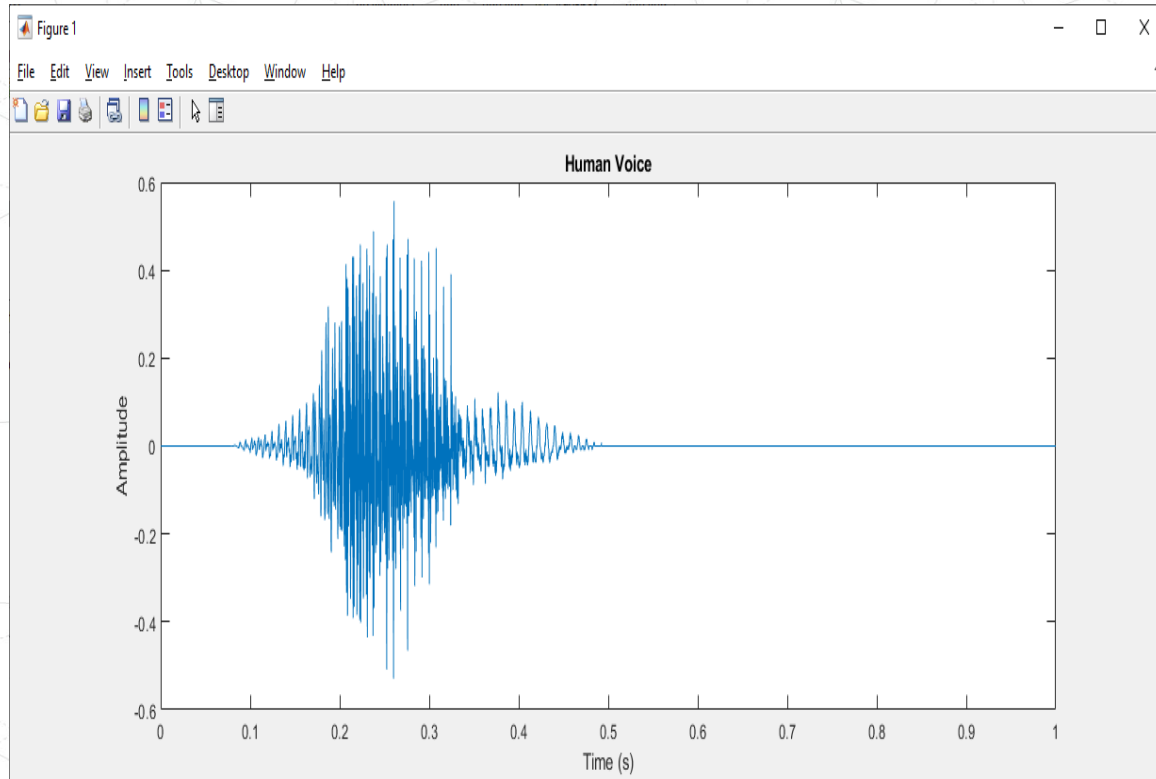
Select a file to view details

Workspace

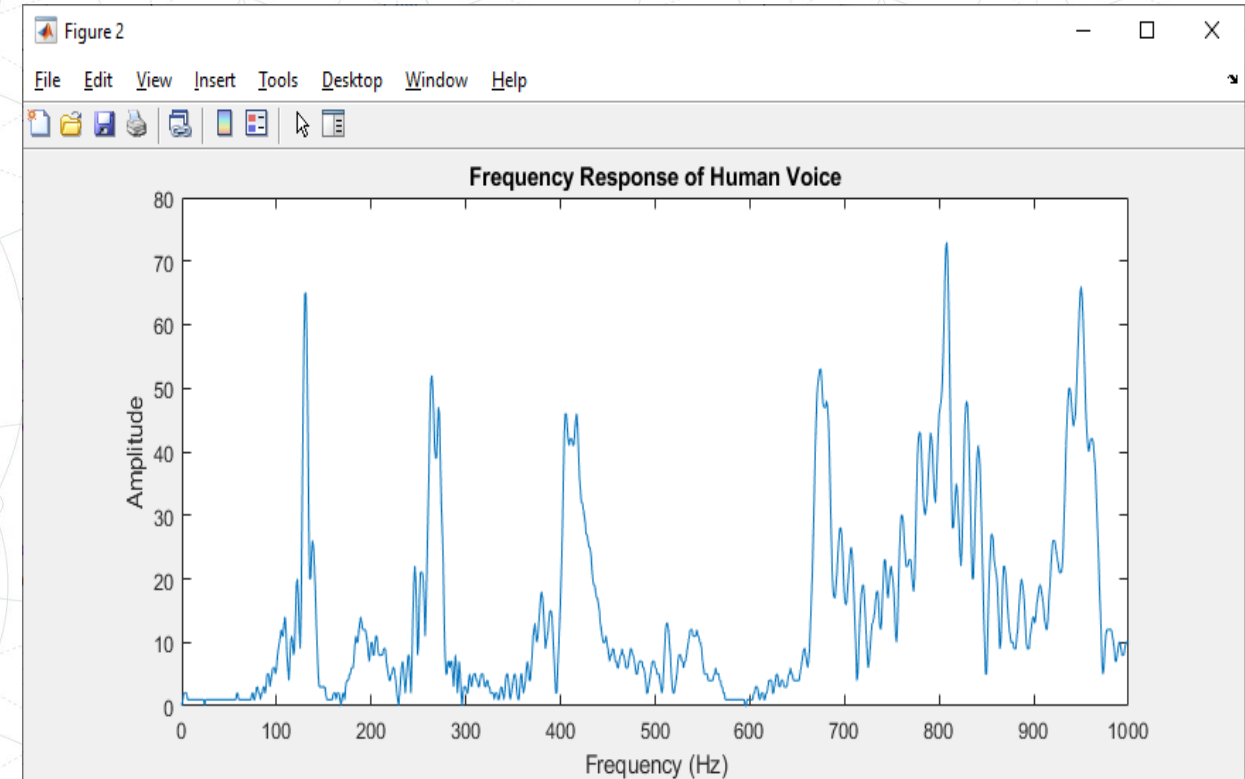
- f
- Fs
- i
- Nsamps
- recObj
- s
- t
- y
- y_fft

```
recorder.m* x +
14 %Do Fourier Transform
15 y_fft = abs(fft(y)); %Retain Magnitude
16 y_fft = y_fft(1:Nsamps/2); %Discard Half of Points
17 f = Fs*(0:Nsamps/2-1)/Nsamps; %Prepare freq data for plot
18 y_fft= int8(y_fft);
19
20 %Plot Voice in Time Domain
21 subplot(2,1,1)
22 plot(t, y)
23 xlim([0 1])
24 xlabel('Time (s)')
25 ylabel('Amplitude')
26 title('Human Voice')
27
28 %Plot Voice File in Frequency Domain
29 subplot(2,1,2)
30 plot(f, y_fft)
31 xlim([0 1000])
32 xlabel('Frequency (Hz)')
33 ylabel('Amplitude')
34 title('Frequency Response of Human Voice')
35
36 %s = serial('/dev/ttyUSB0', 'BaudRate', 9600);
37
38 s = serial('COM6', 'BaudRate', 9600);
39 for i = 1:1000
40     fopen(s);
41     fwrite(s,y_fft(i));
42     fclose(s);
43     disp(i)
44 end
```

Signal Plot



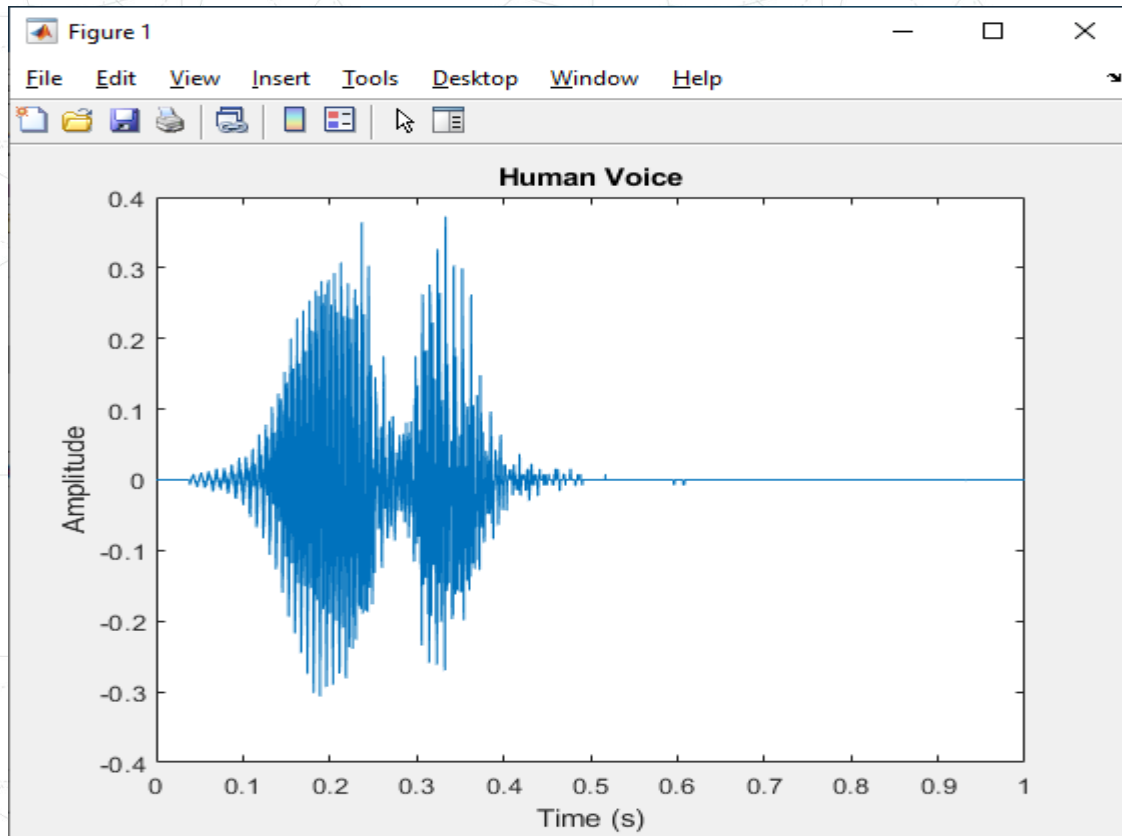
Human Voice Signal



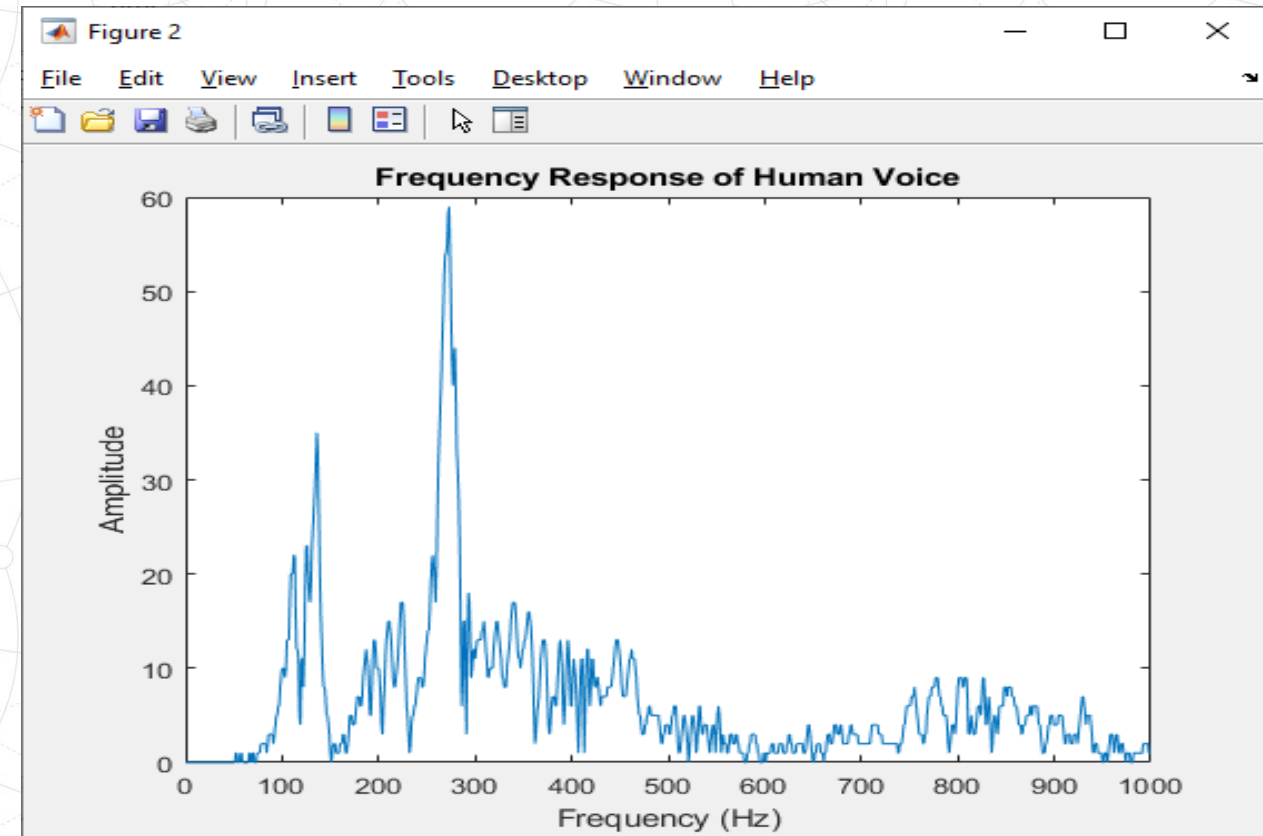
Processed Signal Tx

"ONE" Plotted

Signal Plot



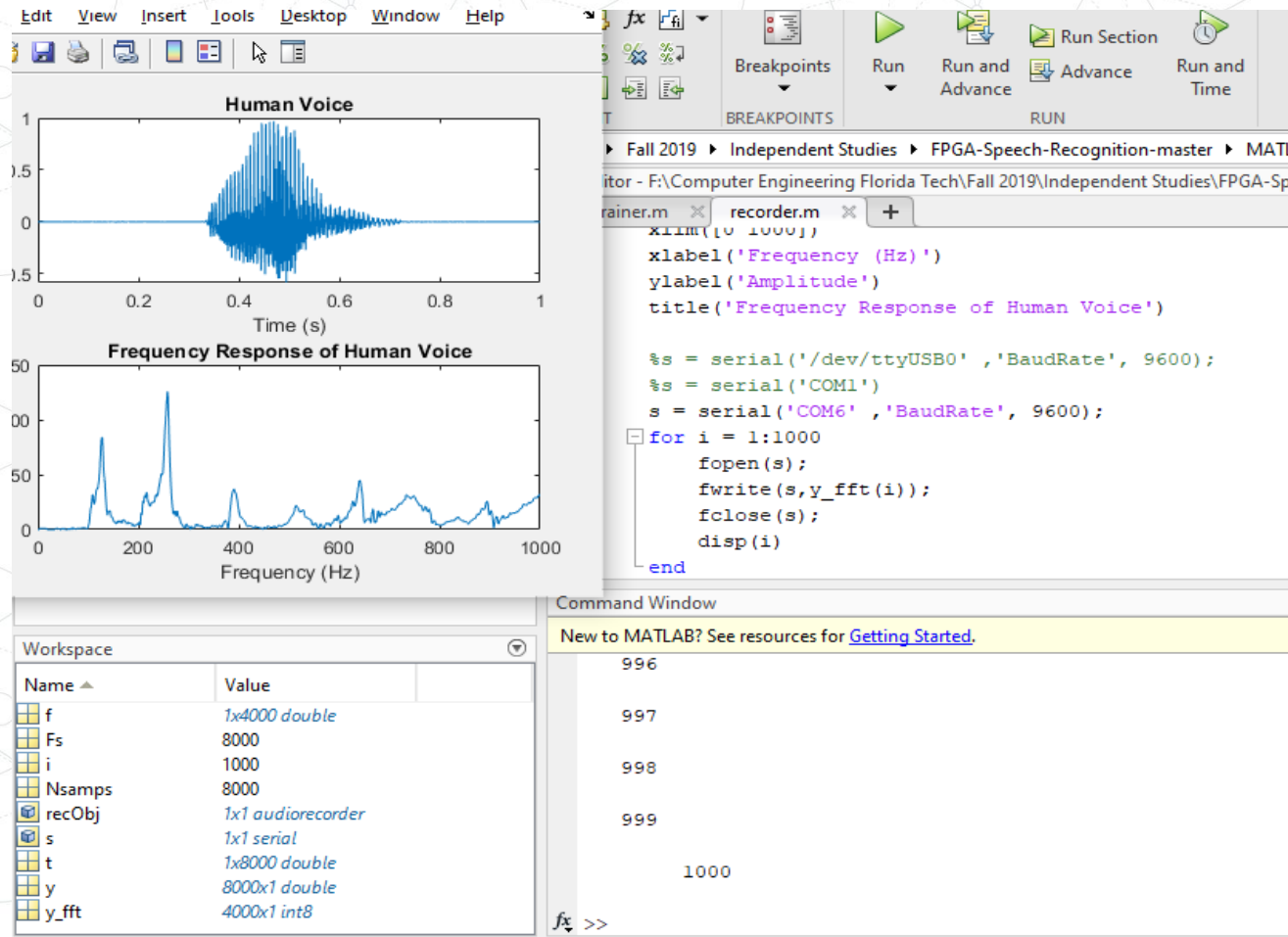
Human Voice Signal



Processed Signal Tx

“ZERO” Plotted

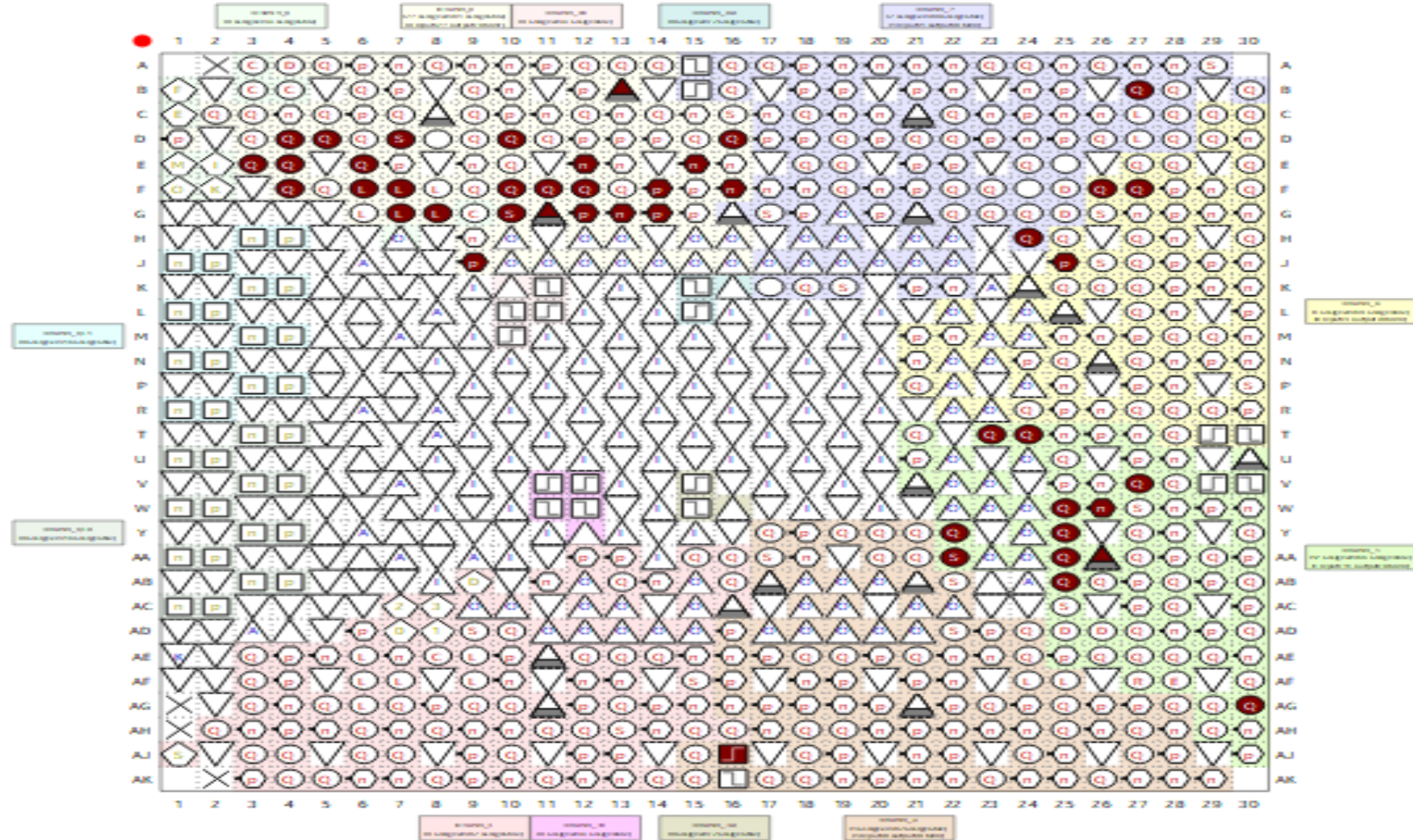
Analysis



Pin Planner Circuit

Top View - Wire Bond

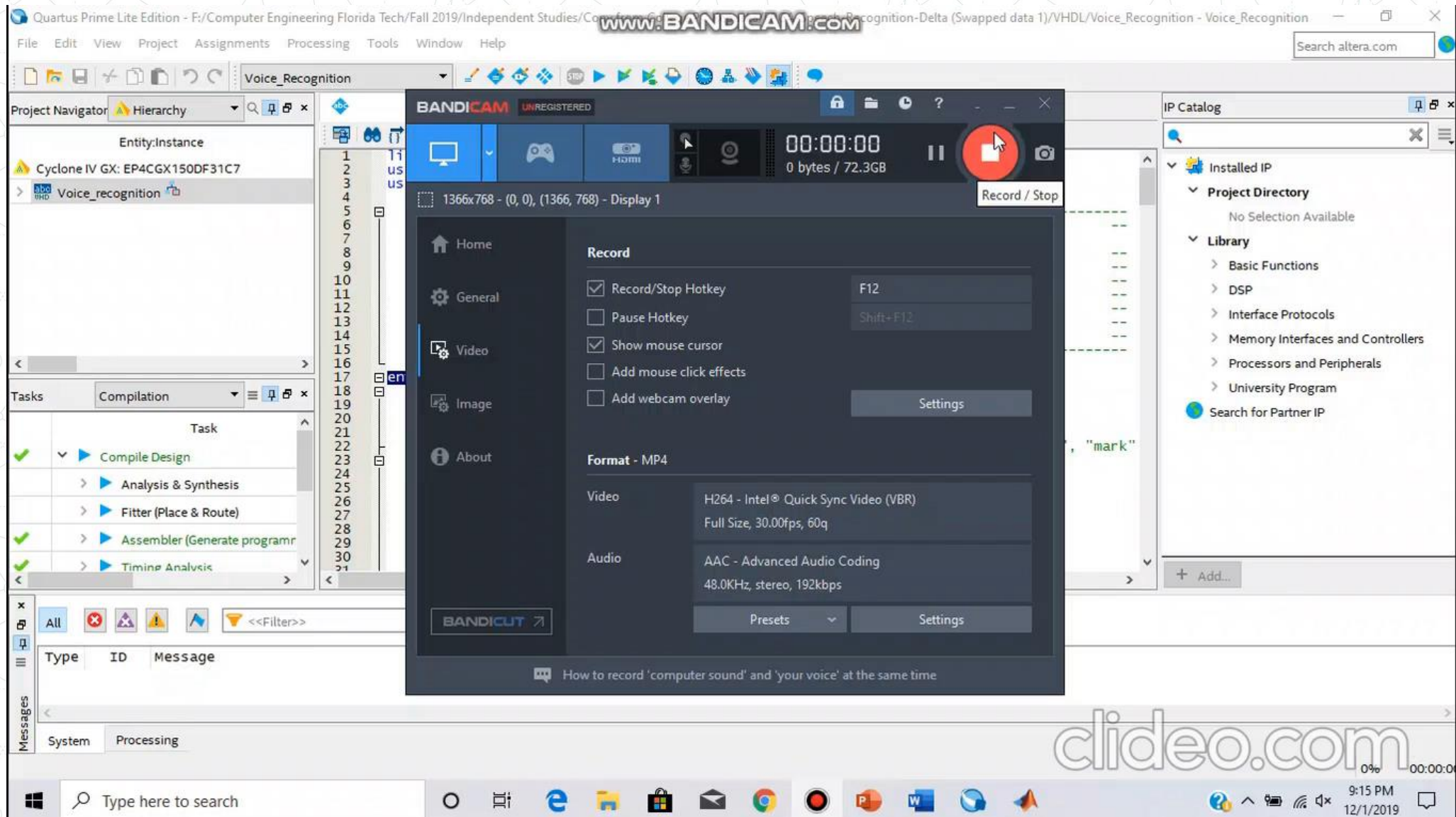
Cyclone IV GX - EP4CGX150DF31C7



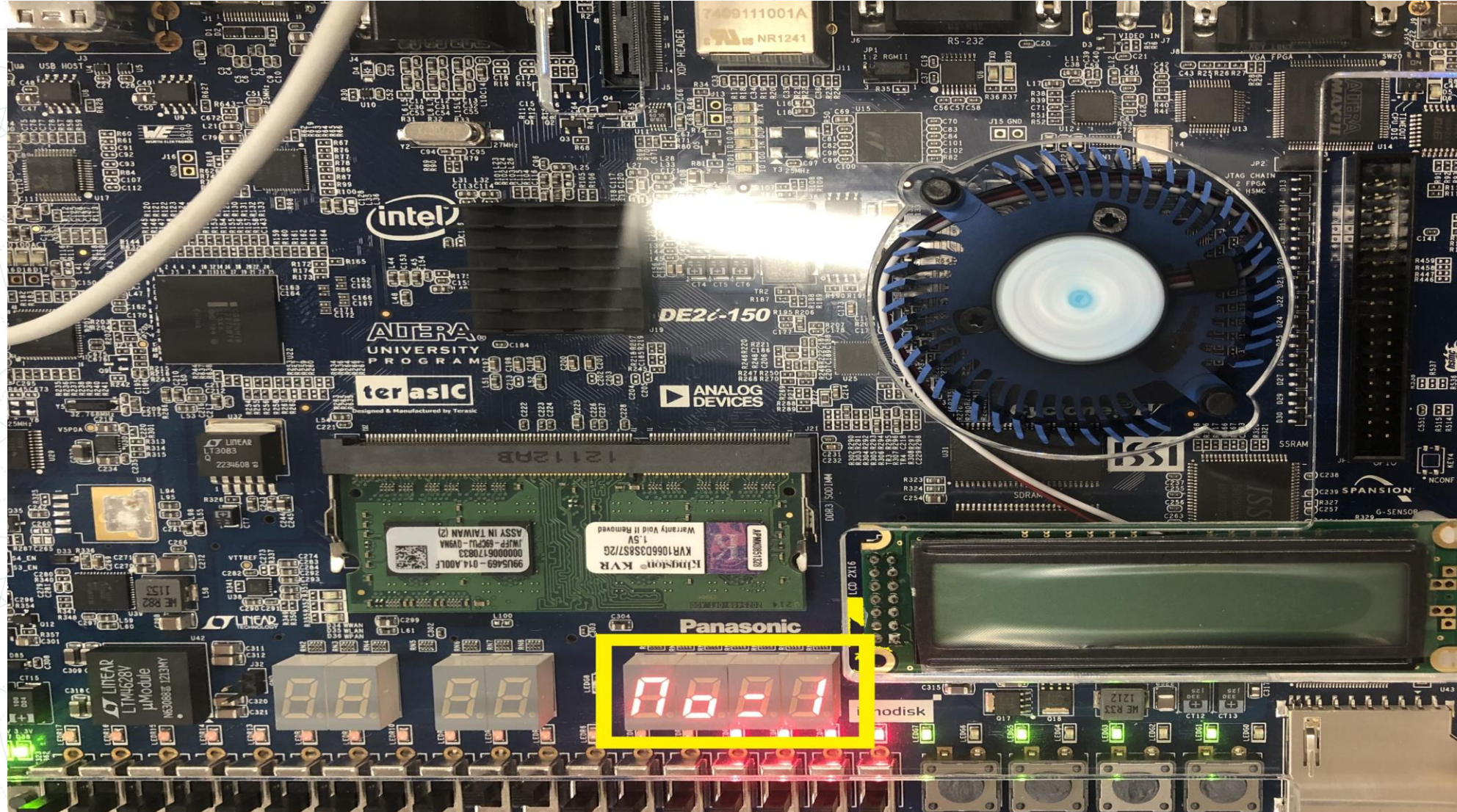
Pin Configuration

Node Name	Direction	Location	I/O Bank	VREF Group	Filter location	I/O Standard	Current	Slew Rate
BUSY	Output	PIN_J25	6	B6_N0	PIN_J25	2.5 V	16mA (default)	2 (default)
CLK	Input	PIN_AJ16	4	B4_N2	PIN_AJ16	2.5 V	16mA (default)	
data_led[7]	Output	PIN_AA22	5	B5_N1	PIN_AA22	2.5 V	16mA (default)	2 (default)
data_led[6]	Output	PIN_Y25	5	B5_N2	PIN_Y25	2.5 V	16mA (default)	2 (default)
data_led[5]	Output	PIN_Y22	5	B5_N1	PIN_Y22	2.5 V	16mA (default)	2 (default)
data_led[4]	Output	PIN_W26	5	B5_N0	PIN_W26	2.5 V	16mA (default)	2 (default)
data_led[3]	Output	PIN_F26	6	B6_N0	PIN_F26	2.5 V	16mA (default)	2 (default)
data_led[2]	Output	PIN_F27	6	B6_N0	PIN_F27	2.5 V	16mA (default)	2 (default)
data_led[1]	Output	PIN_AB25	5	B5_N2	PIN_AB25	2.5 V	16mA (default)	2 (default)
data_led[0]	Output	PIN_AA25	5	B5_N2	PIN_AA25	2.5 V	16mA (default)	2 (default)
equal[6]	Output	PIN_G10	8	B8_N2	PIN_G10	2.5 V	16mA (default)	2 (default)
equal[5]	Output	PIN_J9	8	B8_N2	PIN_J9	2.5 V	16mA (default)	2 (default)
equal[4]	Output	PIN_G12	8	B8_N1	PIN_G12	2.5 V	16mA (default)	2 (default)
equal[3]	Output	PIN_F12	8	B8_N1	PIN_F12	2.5 V	16mA (default)	2 (default)
equal[2]	Output	PIN_G13	8	B8_N0	PIN_G13	2.5 V	16mA (default)	2 (default)
equal[1]	Output	PIN_B13	8	B8_N0	PIN_B13	2.5 V	16mA (default)	2 (default)
equal[0]	Output	PIN_G14	8	B8_N0	PIN_G14	2.5 V	16mA (default)	2 (default)
FRAME_ERR	Output				PIN_L30	2.5 V (default)	16mA (default)	2 (default)
n[6]	Output	PIN_D4	8	B8_N1	PIN_D4	2.5 V	16mA (default)	2 (default)
n[5]	Output	PIN_D5	8	B8_N2	PIN_D5	2.5 V	16mA (default)	2 (default)
n[4]	Output	PIN_E3	8	B8_N2	PIN_E3	2.5 V	16mA (default)	2 (default)
n[3]	Output	PIN_E4	8	B8_N2	PIN_E4	2.5 V	16mA (default)	2 (default)
n[2]	Output	PIN_E6	8	B8_N2	PIN_E6	2.5 V	16mA (default)	2 (default)
n[1]	Output	PIN_D7	8	B8_N1	PIN_D7	2.5 V	16mA (default)	2 (default)
n[0]	Output	PIN_D10	8	B8_N2	PIN_D10	2.5 V	16mA (default)	2 (default)
o[6]	Output	PIN_F10	8	B8_N2	PIN_F10	2.5 V	16mA (default)	2 (default)
o[5]	Output	PIN_F4	8	B8_N2	PIN_F4	2.5 V	16mA (default)	2 (default)
o[4]	Output	PIN_F6	8	B8_N2	PIN_F6	2.5 V	16mA (default)	2 (default)
o[3]	Output	PIN_AG30	5	B5_N2	PIN_AG30	2.5 V	16mA (default)	2 (default)
o[2]	Output	PIN_F7	8	B8_N2	PIN_F7	2.5 V	16mA (default)	2 (default)
o[1]	Output	PIN_G7	8	B8_N2	PIN_G7	2.5 V	16mA (default)	2 (default)
o[0]	Output	PIN_G8	8	B8_N2	PIN_G8	2.5 V	16mA (default)	2 (default)
result[6]	Output	PIN_F14	8	B8_N0	PIN_F14	2.5 V	16mA (default)	2 (default)
result[5]	Output	PIN_D16	8	B8_N0	PIN_D16	2.5 V	16mA (default)	2 (default)
result[4]	Output	PIN_F16	8	B8_N0	PIN_F16	2.5 V	16mA (default)	2 (default)
result[3]	Output	PIN_F11	8	B8_N1	PIN_F11	2.5 V	16mA (default)	2 (default)
result[2]	Output	PIN_G11	8	B8_N1	PIN_G11	2.5 V	16mA (default)	2 (default)
result[1]	Output	PIN_E12	8	B8_N1	PIN_E12	2.5 V	16mA (default)	2 (default)
result[0]	Output	PIN_E15	8	B8_N0	PIN_E15	2.5 V	16mA (default)	2 (default)
RST_N	Input	PIN_AA26	5	B5_N2	PIN_AA26	2.5 V	16mA (default)	
state_four	Output	PIN_W25	5	B5_N0	PIN_W25	2.5 V	16mA (default)	2 (default)
state_one	Output	PIN_T23	5	B5_N0	PIN_T23	2.5 V	16mA (default)	2 (default)
state_three	Output	PIN_V27	5	B5_N0	PIN_V27	2.5 V	16mA (default)	2 (default)
state_two	Output	PIN_T24	5	B5_N0	PIN_T24	2.5 V	16mA (default)	2 (default)
UART_RXD	Input	PIN_B27	7	B7_N0	PIN_B27	2.5 V	16mA (default)	
UART_TXD	Output	PIN_H24	7	B7_N0	PIN_H24	2.5 V	16mA (default)	2 (default)

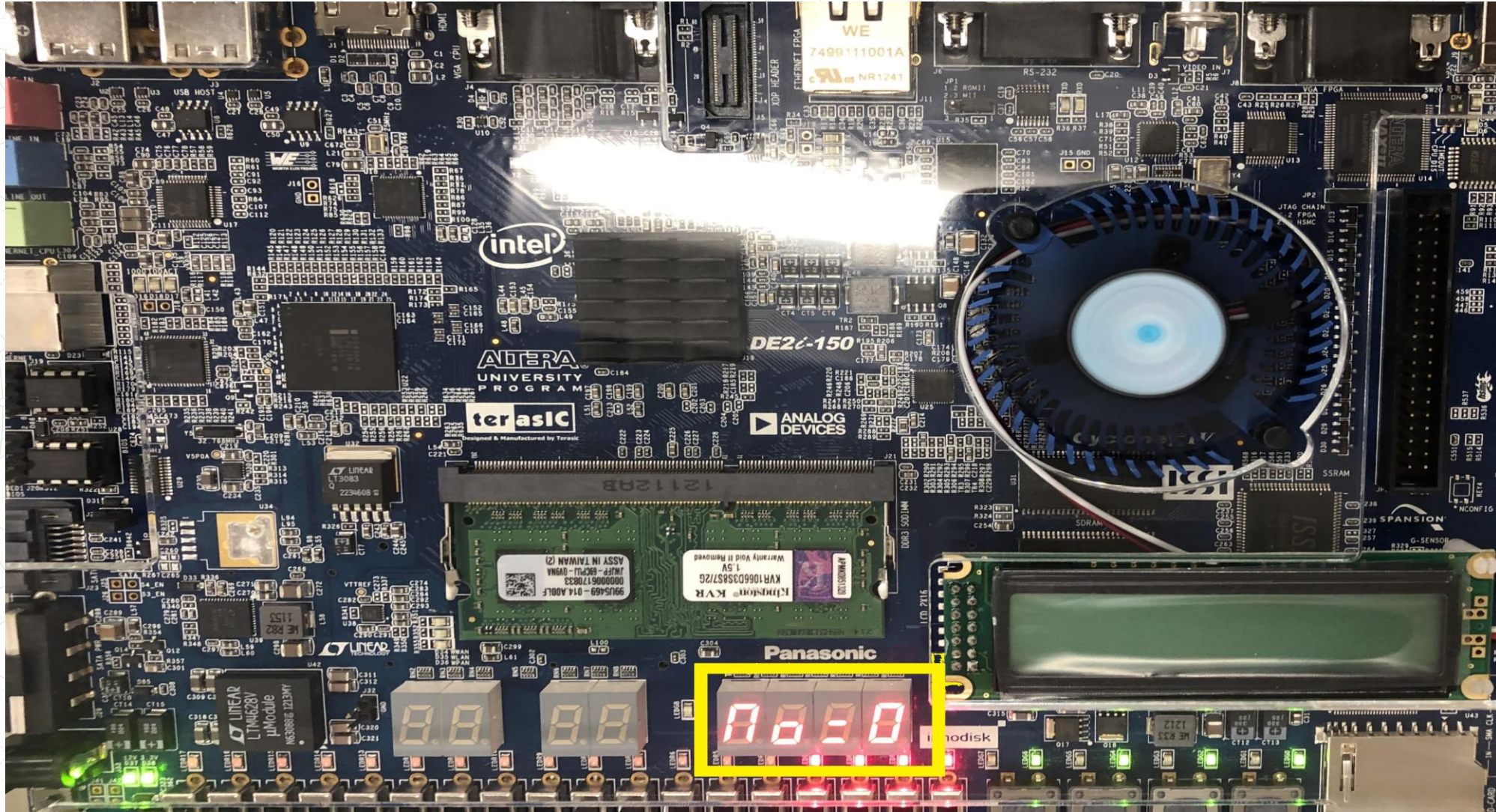
Demonstration for “One”



Result “One” was detected



Result “Zero” was detected



Resources and Results

- RAM consumption approx. 380MB
- Approximately 13,757 Logic Elements Consumed.
- 9144 Registers
- 10,450 Logic Function used
- Accuracy is not static but approx. 60%
- Can detect a speech of either (“One” or “ZERO”)^[6]

Cost

Image	Item	Quantity	Cost
	Computer (MATLAB)	1	-
	DE2i-150 Dev Board	1	700\$
	Mic for Computer	1	20\$
	USB to Serial (RS-232)	1	20\$
		Total	740\$

Conclusion

- Basic speech recognition system on Altera DE2i-150.
- Software was used to overcome many hardware incapability.
- The System was successful in recognizing a (“ONE” OR “ZERO”).
- The test samples to compare and evaluate was few, which intern affects the accuracy of the system.
- The availability of more powerful hardware will make it easier to implement more robust algorithms like Hidden Markov Models and use more powerful ADC Chips to record more pure sound resulting in more accurate results. [6]

References

- [1] “Field-Programmable Gate Array.” Wikipedia, Wikimedia Foundation, 30 Nov. 2019, https://en.wikipedia.org/wiki/Field-programmable_gate_array.
- [2] “Choosing a Wake Word - Tips & Considerations.” Picovoice, <https://picovoice.ai/docs/choose-wake-word/index.html>.
- [3] Kepuska, Veton. “Wake-Up-Word Speech Recognition.” IntechOpen, IntechOpen, 13 June 2011, <https://www.intechopen.com/books/speech-technologies/wake-up-word-speech-recognition>.
- [4] “Altera DE2 Board - Mạch Thí Nghiệm FPGA.” Altera DE2 Board - Mạch Thí Nghiệm FPGA, <http://kitboardmach.blogspot.com/2017/11/altera-de2-board-mach-thi-nghiem-fpga.html>.
- [5] DE2i-150 FPGA Development Kit, 17 May 2019, <https://www.intel.com/content/www/us/en/programmable/solutions/partners/partner-profile/terasic-inc-/board/de2i-150-fpga-development-kit.html>.
- [6] MohammedRashad. “MohammedRashad/FPGA-Speech-Recognition.” GitHub, <https://github.com/MohammedRashad/FPGA-Speech-Recognition>.

Thank you.

Adolf A Dcosta • ECE 5570 • Dec 1st 2019