Automated Fixed Base Operator

Joshua Dean, Michael Graziano, Vanessa Pena, Gilbert Vieux

Dept. of Electrical Engineering and Computer Science, University of Central Florida, Orlando, Florida, 32816-2450

Abstract — **The growth of general aviation continues to expand the number of non-towered airports across the world. This has pushed more airports to build and staff a Fixed Base Operator (FBO) which provides necessary services to airports such as parking, hangar services, mechanical services, flight instruction, pilot lounges, and more. At non-towered airports or during non-towered hours FBOs are sometimes able to provide pilots with local weather, airport information, and radio communication checks depending on staffing. Our system strives to automate this serve of FBOs by providing current local weather comparable to modern ASOS systems and establishing a radio communication check with pilots.**

Index Terms — **Artificial intelligence, audio conditioning, automatic speech recognition, radio communication, weather sensing**

I. Introduction

Before a pilot takes off they should know that their microphone, radio, and headset are operational. This is so that they can communicate with other pilots in the area to avoid deadly collisions and for communicating with Air Traffic Control soon after takeoff. This is necessary if a pilot is planning to fly IFR as they must establish communication with ATC starting on the ground or soon after becoming airborne. Also, as they begin their take off or come into land, a key piece of information is to know the wind direction, wind speed, and gusts at the airport they are taking off or landing at. This is because pilots always need to take off and land into as much as a headwind to increase the amount of wind over the wings to generate lift, increase airspeed, and decrease groundspeed. Current wind information is crucial especially if a crosswind exists, as the pilot needs to choose the best runway to take off or land on. Other weather information such as temperature and the barometric pressure at the airport is also important so that pilots can set their altimeters and judge the density altitude as well as the visibility. Usually an Automated Weather Observing System (ASOS) or an Automatic Terminal Information Service (ATIS) and FBOs are the ones to relay this information as well as other remarks about airport conditions to the pilots over the radio, but some airports do not have a FBOs, ASOS, or ATIS. Furthermore, most FBOs are not staffed 24 hours a day throughout the year. One solution to try to mitigate this issue at such airports is a windsock, which is a light and flexible cone of fabric mounted on a mast, usually somewhere along the airstrip of an airport. Windsocks let the pilots know some of the important weather readings, such as wind direction, but they are small and cannot be seen until the aircraft is very close to the airport. On the other hand, there are some automated systems currently on the market that perform task such as broadcasting weather conditions and transmit radio checks, but they are costly and not suited for smaller airports.

The Automated Fixed Base Operator is a low-cost system that satisfies these two basic needs. This system broadcasts important weather information when prompted by pilots in the area. For example, when the system is prompted, the system will broadcast a weather report that includes the latest recorded wind direction and wind speed as well as gusts, temperature, dew point, density altitude, and airport remarks. This system also performs a transmit radio check for any pilot that consists of recording the transmission from the pilot and playing it back along with the power level so the pilot knows exactly how operational their equipment is. Therefore, this system can be classified as an Automated Fixed Base Operator for small airports. The Automated Fixed Base Operator would act as a hub of communication for these small airports that do not have a dedicated FBO or weather station, as well as FBOs and airports who wish to automate this service fulltime. This system provides a source from which any pilot can obtain crucial weather information or perform any radio communication checks they need prior to taking off and landing their aircrafts.

Our goal in the design of this Auto FBO to connect a weather station and VHF radio through an interface board to a microprocessor that can process all the necessary information required to be comparable to modern ATIS and ASOS systems, as well as preform quality radio communication checks. Using these components, we build a system that can assist pilots in taking off, flying, and landing safely, while being configurable and cost-effective.

II. System Overview

The overall requirements of our system are that it will be able to recognize a microphone click cadence signal from a pilot and decide from the cadence if the pilot is requesting weather condition information or a communications check. If the pilot is requesting weather information, the system will respond with an ATIS style broadcast with the wind speed, direction, temperature, dew point, pressure, density altitude, and airport remarks. If the pilot is requesting a communications check, the system will respond with a message acknowledging the request and will record and playback the pilot's response so they can hear exactly how their message was received. The system will also respond with a power level to inform the pilot of their signal strength.

The secondary requirements of the system are to have a web graphical interface from which the user can read the current weather and make parameter changes. The Auto FBO system is to operative on airports UNICOM frequency and will not broadcast if the channel is being used. If the wind conditions change the system will broadcast the current wind conditions if they change such that they exceed the chosen parameters. The Auto FBO system will also announce a favorable runway if conditions are met within chosen parameters. Upon receiving the designated cadence for a communications check, the system will record the pilot's transmission and sequentially transmit the recording with no added distortion or attenuation. After the playback, the recording will also playback the power level received. Shown in figure is a table of the engineering specifications set for the Auto FBO system.

Response Time	< 3s
Temperature Accuracy	± 2°C
Humidity Accuracy	± 5%
Pressure Accuracy	± 0.12 inHg
Wind Direction Accuracy	± 5°
Wind Speed Accuracy	± 2 kts or ± 5%
Recording Length	≤ 15s
Power Level Accuracy	± 5 dBm

Fig. 1. Engineering specifications of the Auto FBO system.

A. Technical Overview

Shown below in figure 2 is the overall block diagram of the system. Exiting the aviation radio is the squelch and AGC (Automatic Gain Control) signals, along with the receive audio signal. The squelch signal is directed to the squelch detect system which relays to the Raspberry Pi (RPI) if there is another radio transmitting on the same frequency as the aviation radio. This information is given to the RPI via GPIO. The AGC signal is first sent to the ADC and then sent via I2C to the RPI. The AGC signal from the radio is known to vary from 3.5V to 1.28V which corresponds to the signal power the radio is receiving from another radio transmission. This correlation is shown in figure 3. The ADC translates this analog voltage to a digital signal, which the RPI will reference to the known correlation shown in figure 3. The received audio signal is first passed through the audio conditioning system and then relayed to the CODEC which transfers the audio signal digitally to the RPI via I2C.

Coming into the radio are the PTT (Push to Talk) and transmit audio signals. The PTT system receives 3.3V from the RPI whenever the Auto FBO wants to broadcast through the radio via a GPIO pin. The PTT will then pull the radio's PTT to ground so that the system can broadcast. Once this action is complete the audio signal the system wants to broadcast is pushed through the CODEC and audio conditioning system from the RPI. The CODEC will then translate the digital audio signal to an analog signal, afterwards the audio signal is conditioned to be given to the radio.

The two weather sensors at the top of the system block diagram first send their signals to the interface board which is then relayed to the RPI. The anemometer signals are first given to the ADC to be transformed into digital data via I2C. The THD (Temperature Humidity Dewpoint) sensor, however, communicates directly to the RPI via the interface board.

The power supply unit on the interface board provides 3 voltage rails for the Auto FBO system of 15V, 5V, and 3.3V. This system receives power from a commercial AC/DC power supply and creates the voltage rails with regulators.

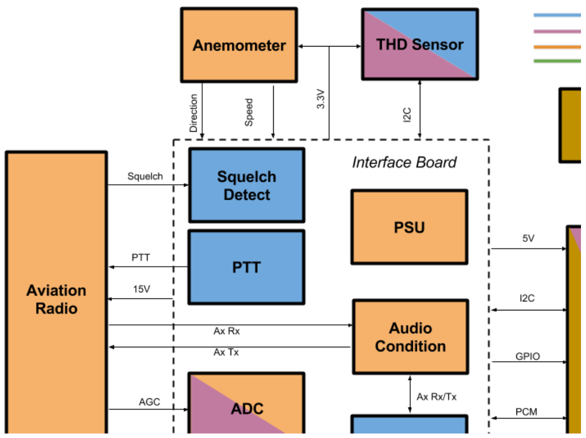

Fig. 2. System block diagram of the Auto FBO System with design responsibilities of each member.

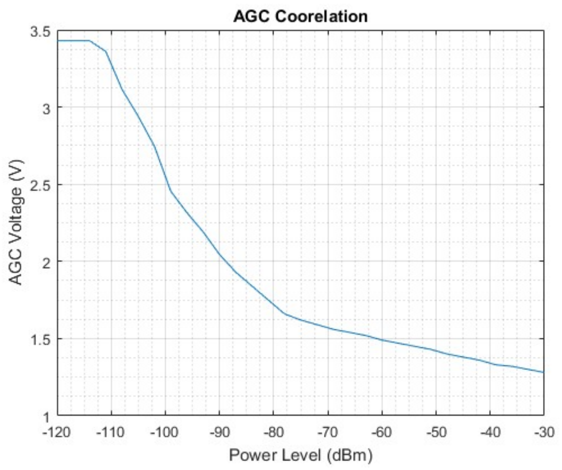

Fig. 3. AGC voltage to received power level from aviation radio. This correlation is only valid for the specific radio used in our Auto FBO system, as each radio will have difference voltages ranges and correlations.

III. Major System Components

This section covers the major components used in the Auto FBO system including the Raspberry Pi 3 Model B, ADC, aviation radio, THD sensor, and operational amplifiers.

.

A. Raspberry Pi 3 Model B

The two microcontrollers deliberated over were the Raspberry Pi 3 Model B and the Arduino Uno. The Raspberry Pi was the clear winner for our project; the Pi was favored not only because of its specifications, but also because the team members had more experience with this specific microcomputer. We researched both microprocessors thoroughly before finalizing our decision; we chose the Pi because of its versatility, accessibility, and open-source libraries. One slight problem was that the Pi lacks an analog-to-digital converter which is needed to process the incoming analog signals from our sensors; on the other hand, the Arduino has a built-in A/D converter while the Pi isn't naturally equipped with one; but, that did not really impact our decision as much because we made use of an external A/D converter paired with our MCU.

One of the reasons we selected the Pi is because of its naturally optimized operating system called Raspbian. This Linux-like operating system is distributed with over 35,000 packages and pre-compiled software bundle meant to improve the Pi. It also makes its overall installation process as well as interfacing with peripheral devices quite easy. The programming experience is made simpler by providing a graphical interface to the user. Raspbian is a fully-fledged Linux-based operating system used by the Pi (which in turn is basically a small computer) as stated above, but the Arduino Uno is only a microcontroller. Using the Raspberry Pi 3 as a basic Linux computer allows us to possibly set up a graphical interface in the future, while also providing us with a headless command setup now. The Arduino Uno still supports many functions required by our project. This includes the key function of receiving and converting inputs from sources such as a temperature sensor or anemometer using its built-in A/D converter. Unfortunately, it also does not support a multitude of specifications required by our project such as Wi-Fi access or python.

The Arduino Uno does not provide the user with a variety of coding languages. IDLE's are not compatible (as shown in figure *3.2.1.3*) with Arduino; instead, the user is provided with specifically designed tools to setup and program the different Arduino models. The codes written on the board are known as sketches and are written in C++. This was one of the main deal breakers that pushed our decision towards the more favorable Raspberry Pi. We selected python as our coding language for the ability to interact with Django -a database framework that allows us to store data on the Pi. Also, python offers many packages to deal with analog signals which further narrowed down our choice of coding languages.

Furthermore, the Raspberry Pi includes a faster processor (running at 2.4 GHz), multi-tasking power (as opposed to Arduino's focus on running one simple program), and it is an independent computer (Arduino Uno is not). The onboard Ethernet network card, the wireless capability, and the graphical interface provided by the Pi shows its superiority with software applications and usability. This graphical interface is an imperative requirement as our sponsor mentioned his desire to change some of the functionalities implemented by our project; such as, changing the current airport location easily or the click-pattern. Also, access to the internet via Wireless Lan or Ethernet connection is required to communicate to our web interface. Another feature on the Raspberry Pi 3 that contributed to its selection is the 2.4GHz 802.11n wireless capabilities and the 10/100 Ethernet port.

B. Aviation Radio

The radio chosen to be used was the ICOM IC-A2 VHF radio. It is a compact, synthesized, 5W PEP, VHF transceiver. The IC-A2 offers keyboard frequency selection with extremely good stability and frequency accuracy. What lead our team to choose this radio was that we needed an older radio that we could open up and solder wires to its PCB, along with a detailed schematic and manual so that we could find where its AGC, squelch, PTT, and audio components were located.

C. THD Sensor

Among all the temperature, humidity, and pressure sensors the chosen device to cover these measurements was the MS860702BA1. Not only was it chosen because it could be used for temperature, humidity, and pressure measurements, but also its specifications compared to the other parts. Also, in terms of price it is clearly a better selection, especially if mass production of this system is to be implemented.

The MS8607 is the novel digital combination sensor of MEAS providing 3 environmental physical measurements all-in-one: pressure, humidity and temperature (PHT). This product is optimal for applications in which key requirements such as ultra-low power consumption, high PHT accuracy and compactness are critical. High pressure resolution combined with high PHT linearity makes the MS8607 an ideal candidate for environmental monitoring and altimeter in smart phones and tablet PC, as well as PHT applications such as HVAC and weather stations. This new sensor module generation is based on leading MEMS technologies and latest benefits from Measurement Specialties proven experience and know-how in high volume manufacturing of sensor modules, which has been widely used for over a decade.

Regarding its temperature measurements, the MS860702BA1 performs best among the other parts in max response time and power consumption. Its temperature range is third best; however, its range is more than adequate. The accuracy of the device is the worst among the selected devices, but is sufficient enough for accurate weather reporting. Resolution is among the best, along with its long-term stability. The humidity and pressure specifications of the device is overall the best out of all the possible selections.

The MS8607 includes two sensors with distinctive MEMS technologies to measure pressure, humidity and temperature. The first sensor is a piezo-resistive sensor providing pressure and temperature. The second sensor is a capacitive type humidity sensor providing relative humidity. Each sensor is interfaced to a $\Delta\Sigma$ ADC integrated
circuit for the digital conversion. The MS8607 converts both analog output voltages to a 24-bit digital value for the pressure and temperature measurements, and a 12-bit digital value for the relative humidity measurement.

Another reason this sensor was selected was because it can be communicated with via I2C. Since the anemometer uses the same communication protocol, it greatly simplifies integration if both sensors run on the same protocol. The external microcontroller clocks in the data through the input SCL (Serial Clock) and SDA (Serial Data). Both sensors respond on the same pin SDA which is bidirectional for the I2C bus interface. Two distinct I2C addresses are used (one for pressure and temperature, the other for relative humidity). The I2C address for pressure and temperature is 1110110, while the I2C address for humidity is 1000000.

D. Operational Amplifiers

The usage of operational amplifiers in the Auto FBO system are exclusively for audio signals. These signals have a bandwidth of roughly 0 to 20kHz. A quality operational amplifier will have the basic requirements of low noise, low total harmonic distortion (THD), good response (slew rate), and low power. However, these are somewhat conflicting requirements. Typically, lower power operational amplifiers with have poor noise and THD specifications.

The parameters chosen to compare operational amplifiers for the design were noise, slew rate, gain bandwidth product, total harmonic distortion, supply voltage and current, CMRR, and price per unit. The main factors in this comparison are noise, THD, and slew rate. During the recording and playback of the voice communication check, our system strives to not change the incoming audio signal in any way. Thus, low noise and THD is needed along with a good response rate. The GDP, CMRR, and supply voltage and current are also important features to the characterization of an operational amplifier, and were thus included. Cost is also of concern as our goal is to produce a low-cost product. However, a higher performance device will obviously cost a lot more.

The chosen operational amplifier was the NE5534A. It was determined that the needed slew rate for audio signals up to 20 kHz was 0.377 µs/V. This was determined by the equation SR = 2πfV where f is the maximum frequency of interest and V is the max voltage. This slew rate was met by all the chosen candidate operational amplifiers, but some overhead was preferable. The NE5524A also has a great noise figure even comparable with the high performance OPA models. These factors along with its other specifications and low price is why this operational amplifier was chosen.

IV. Hardware Design

This section will cover the design of the power supply, PTT, carrier detect, audio, and weather sensor design.

A. Power Supply Design

Shown in figure 4 is the schematic for the power supply design. The central power supply unit (CPSU) supplies 20V to all four linear voltage regulators. This is a commercially bought AC to DC power supply. The line to the regulators also contains shunt electrolytic capacitors. These capacitors are included for several reasons including recommended application suggestions of the datasheets, increased capacitance, low ESR, high frequency impedance, reliability, redundancy, and peak current demands. The output of each regulator also includes shunt electrolytic capacitors for the same reasons. KEMET 49X tantalum capacitors were used for the input and output capacitors to aid in the design in respect to the characteristics above.

The input pin of the 5V switching voltage regulator, LM2676-5.0, is supplied power by the 20V central power unit to regulate a fixed output voltage of 5V at the output pin. The input voltage of 20V from the central power unit is acceptable since the device has an absolute maximum input voltage of 45V. The output circuitry of this regulator was designed with guidance from the LM2676-5.0 datasheet. The 100 µF capacitors are used to smooth the switched DC output voltage and provide energy storage for peak supply demands. The 33 µH inductor was chosen to efficiently store energy during the on-switch time and transfer its stored energy during the off-switch time. The 1N5822 catch diode provides a current flow path when during the off-switch time, when the current through the inductor continues to flow. During this time, the diode is forward biased and clamps the switch output to a voltage below ground. The efficiency of the supply is significantly impacted by the power loss in the diode. During the on-switch time the diode is reversed biased. “The boost capacitor creates a voltage used to overdrive the gate of the internal power MOSFET. This improves efficiency by minimizing the on resistance of the switch and associated power loss.”

The linear voltage regulators have their input pins connected to the 20V supply with the exception of the LT1129-3.0, in which its input is connected to the 5V regulator to reduce power loss in the device. The LT1129-3.0 also has a feedback sense pin tied to its output as well as an TVS for safety since that supply we be outside with no casing or shielding on the anemometer.

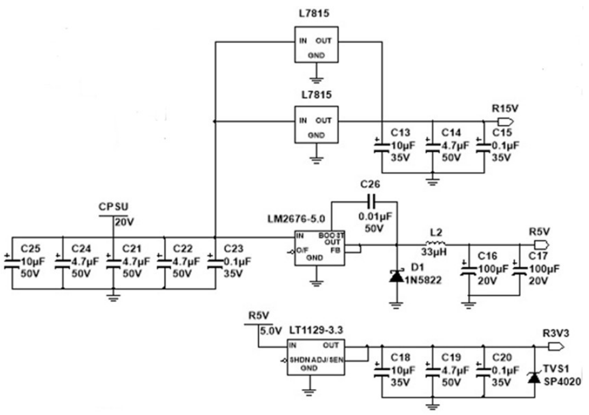

Fig. 4. The schematic of the power supply design.

B. PTT and Carrier Detect Design

In order for the Raspberry Pi to communicate on the channel that the radio is tuned to, the push to talk switch on the radio needs to be pulled to ground. Thus, our PTT system needs to pull the radios PTT line to ground by use a GPIO pin on the RPI. This is done by using a BJT transistor with a flyback diode to protect against inductive loading. When the GPIO is low the PTT line of the radio is open from ground, and when it is high the line is shorted to ground through the BJT transistor.

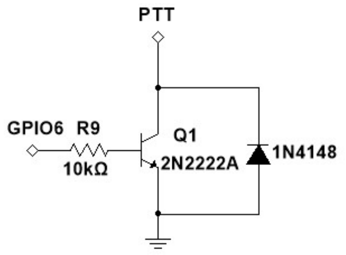

Fig. 5. The schematic of the PTT system.

The goal of the carrier detect system is to relay to the RPI when the channel that the radio is tuned to is being used. This is done through the usage of a comparator. The comparator being used is the LM393 Dual Differential Comparator. The purpose of this device is to compare two voltage values, and output a digital signal indicating which of the two is larger to the RPI through a GPIO.

The differential comparator consists of a high gain differential amplifier. These devices are commonly used in systems that measure and digitize analog signals such as analog to digital converters, as well as relaxation oscillators. In our application, we compare the received signal, carrier detect present or carrier detect not present, with a reference voltage.

For the radio used in our system a 1.4V signal was present on the squelch line when there was a carrier on the channel. This allows us to set a reference voltage of 1V on the comparator. If there is a 1.4V signal on the squelch the comparator will output a logical 1 (3.3V), and if there is no squelch signal the comparator will output a logical 0 (0V). The 1V for reference are achieved through a voltage divider circuit. The input (Vin-) is 5V which is then divided through both resistors of 1k ohms and 250 ohms. The reference is then then compared to the ground at the 1k ohms resistor. This will create a constant output of 1V since the 5V is being provided by a voltage regulator.

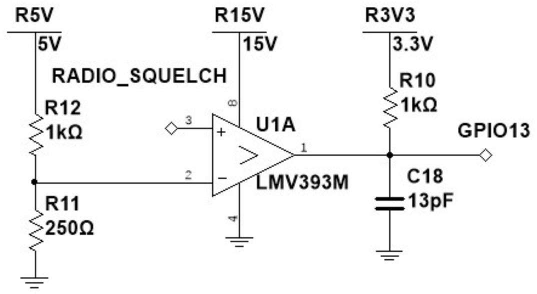

Fig. 6. The schematic of the carrier detect system.

C. Audio Conditioning Design

In transmitting audio out from the Raspberry Pi then to the CODEC we found that there was high frequency noise being produced. This lead to the decision of a low pass filter being needed. This was done by using a second order low pass Butterworth filter with a cutoff frequency of 30 kHz. The Butterworth filter was chosen as it can provide a maximally flat passband which is needed as to not alter the audio signal. The cutoff frequency of 30 kHz was chosen since it is known that audio signals range from 0-20 kHz, and to ensure that as the passband started to drop off near the cutoff frequency it would produce negligible difference on the upper band audio signals.

Using frequency scaling kf was set to 2π×30000 to set the cutoff frequency at 30 kHz. C1 was set to 470 pF to set C2 235 pF so that these capacitors could be commercially bought. Solving for the magnitude scaling factor sets R to 16 kΩ.

$$C'_1 = \frac{1}{k_m k_f}\frac{2\sqrt{2}}{2} = 470 \text{ pF} \ (1)$$

$$C'_2 = \frac{1}{k_m k_f}\frac{\sqrt{2}}{2} = 235 \text{ pF} \ (2)$$

$$R' = k_m = 16 \text{ k}\Omega \ (3)$$

Before this filter is a decoupled inverting amplifier circuit network of unity gain which sets an offset of 7.5 V since the operational amplifiers are set between 15 V and ground. Without this network, the audio signal could potentially be cut off. The resistor divider biasing technique is low in cost and keeps the op-amp's dc output voltage at halfway between the supply voltage. The gain of the inverting amplifier circuit is set to one by having the resistor values equal to each other. The high pass filter DC blocking sections at the beginning and end of this audio chain are used to block DC and discharge the blocking capacitors as to not build charge. The cutoff frequency of the blocking circuits is approximately 0 Hz as to not interfere with the audio signals.

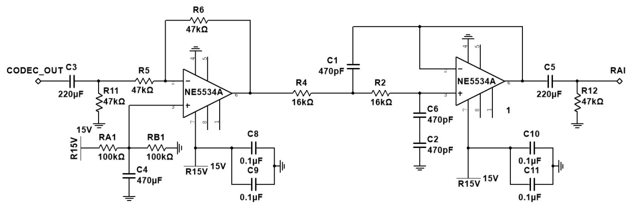

Fig. 7. The schematic of the CODEC output to radio input audio condition system.

While testing the radio with the CODEC it was found that the CODEC was severely loading the radio when it was transmitting audio to the CODEC while the CODEC was recording. To negate this problem a buffer was inserted between the audio signal coming out of the radio and the input of the CODEC.

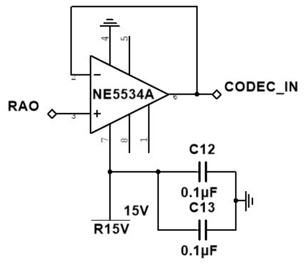

Fig. 8. The schematic of the radio output to CODEC input audio condition system.

D. Weather Sensor Design

For the THD sensor a local 0.1µF capacitor to the VCC and ground pin, as well as join its I2C lines to the main I2C bus. The anemometer hardware design required a 10kΩ resistor on the output of the wind speed line to limit the amount of current though its reed switch, which closed after its wind speed cups completed one cycle. Two TVS were also tied to the anemometers wind speed and direction lines for safety.

V. Software Design

After the Raspberry Pi is powered on, it will automatically launch the main program. After the main program is started, it will begin an initialization process. This process includes starting the separate weather program and making sure it is operational and responsive, then it will also start the webserver. The separate weather program will poll the sensors for wind speed, direction, temperature, humidity, and pressure and when called upon, it will return an object will the most current values for each weather condition.

We chose to separate this into its own program because it allows the main program to handle the carrier detect more efficiently, allows us to easily compute the wind speed and direction values, and it will also allow us to set intervals for how often we want certain weather conditions to be read or computed without overcomplicating the main loop. It will be much more efficient to receive an object with all the weather readings in the main program instead of having to poll each sensor when the information is requested. Polling each sensor when the weather is requested would result in a delay of when the synthesized audio would play back to the pilot. This is due to the nature of some of the sensors and the measurements being read. In order to report wind speed and direction, the values have to be calculated by recording values from the anemometer over a period of time and then finding the average. In addition, the temperature/pressure/humidity sensor has a delay of a couple of seconds while it takes its measurements.

After the initialization process, the main program will begin to listen for a carrier signal. When a carrier signal is detected, the program will enter a function to count the clicks which is described in detail in section 5.2.2 of this document. After the clicks are detected and a decision is made as to whether the pilot is requesting the weather or a communications check, the main loop will jump into either function and perform the needed action.

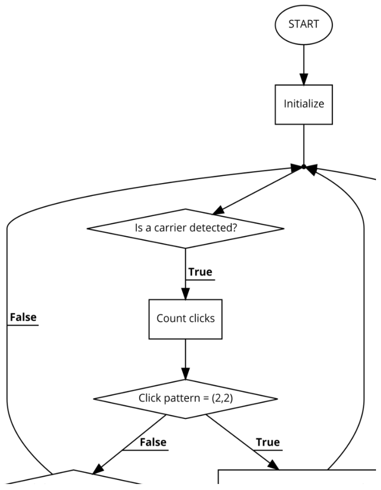

Fig. 9. The main logic loop of the system's software.

The poll weather conditions process is the side process which is started by the main program during its initialization. This process will do all the communicating with the weather sensors and will read and store their values into an object that the main program will request whenever a weather request signal is detected. The program starts by verifying that it can communicate with all the sensors and then it will reset all of its temporary variables. Then it will enter the infinite loop where it polls and stores the readings from each sensor. The temperature/pressure/humidity sensor will only be accessed on a timer because the sample from that sensor has a slight delay and the weather conditions it reads do not change very often.

The count clicks process counts the number of times the pilot keys their radio and checks the duration of each click or spacing to be sure the program is not picking up accidental clicks or the wrong signals. This process is triggered whenever the main program encounters a click. This process will then time the click, considered the "on" time, and if it falls between specified maximum and minimum parameters, the program will move on to the "dwell" time which is the spacing between clicks. It will continue to check each segment according to the patterns we have designated for a communication check or weather report until either a duration does not fall between the specified parameters or we don't receive the segment we were expecting. After the second click or "on", the program will time a "gap" instead of a "dwell" which has a longer duration in order to register a pause between the first sequence of clicks and the second sequence of clicks. If this pause, or any duration, does not fall between the specified parameters, the process will end, ignore the accessed clicks, and will start over to listen for the next new click. Once either pattern sequence is found, the program will decide and escape into the corresponding function to report either a communication check or the weather.

The transmit weather conditions process will start after the main function recognizes the click pattern for weather reporting. From there, the main program will make a call to the weather polling process to receive the current weather object. Then the program will separate each piece of the object, collect all the voice files needed to synthesize each condition, and then concatenate them into a single audio file. Once it has created the audio file for the current weather conditions, it will check the transmission line to ensure that the playback does not step on anyone. After it has checked that the line is clear, it will then broadcast the current weather conditions for the airport including wind speed, wind direction, gusts, temperature, pressure, and humidity. This process must be efficient so that there is not a noticeable or substantial delay between the time that the pilot clicks their radio and the time that the weather report starts to transmit.

The radio communications check process will start after the main function recognizes the click pattern as the correct pattern for a communications check. After it has made the decision to proceed with a communications check, the program will check the transmission line to make sure no one else is on the line. Once the line is clear, then it will transmit a prompt to the pilot which will acknowledge their request for a communications check and ask them to proceed with their transmission. As soon as the next carrier is detected, the program will begin recording the audio transmitted and it will stop recording when the carrier is no longer detected. As discussed earlier in this document, we will be using an audio codec which will allow us to efficiently record audio directly into a WAV file to easily play back. This reduces a lot of overhead since we do not have to create the WAV file manually. After the carrier signal is no longer detected, the program will again check to make sure the line is clear and then it will transmit the recorded audio file back to the pilot. After the audio file, the program will also announce a signal strength level based off the recording. This will allow the pilot to get a better idea of the quality of their transmissions and allow them to make adjustments as needed.

ACKNOWLEDGEMENT

The authors wish to acknowledge the assistance and support of Dr. Lei Wei. They would also like the thank Mr. Michael Young for his advice and guidance, as well as allowing the group to use his facilities.

REFERENCES

[1] TE Connectivity Measurement Specialties. (2017) MS8607-02BA01 Data Sheet. Retrieved July 14, 2017. World Wide Web: http://www.te.com/commerce/DocumentDelivery/DDEController?Action=srchrtrv&DocNm=MS8607-02BA01&DocType=DS&DocLang=English

[2] Texas Instruments. (2014) NE5534X Data Sheet. Retrieved July 14, 2017. World Wide Web: http://www.ti.com/lit/ds/symlink/ne5534a.pdf

[3] ICOM. (2001) IC-A2 Maintenance Manual. Retrieved July 14, 2017. World Wide Web: https://www.manualslib.com/products/Icom-Ic-A2-3679764.html

[4] Texas Instruments. (2016) LM2676 Data Sheet. Retrieved July 14, 2017.World Wide Web: http://www.ti.com/lit/ds/symlink/lm2676.pdf