**Full System Description of the Magnetic Loop Antenna (MLA) Control System**

This system is designed to automate the tuning of a **Magnetic Loop Antenna (MLA)** for a given **VFO frequency** using two **ESP32 microcontrollers**—the **Master ESP32** and the **Slave ESP32**—in conjunction with an **Icom IC 705 transceiver** and a **NanoVNA**.

Xx: I would like to highlight that the NanoVNA will only be used for the characterization not to make the user believe that he will need a nano vna connected somewhere during operation… same for the pc….

The system allows for precise antenna tuning without the need for constant PC involvement after the initial configuration phase.

**Key Components and Roles:**

1. **ESP32 Master (Station - STA)**:
   * Acts as the primary controller and user interface via a **TFT touchscreen**.  
       
     xx: its totally wireless, no cables , i.e. “Master” is a stand alone little box hosuing an ESP and a TFT
   * Establishes a **Bluetooth connection** with the **Icom IC 705 transceiver** to retrieve the current **VFO (Variable Frequency Oscillator)** frequency and display it on the screen.
   * Sends queries to the **ESP32 Slave** to calculate the correct stepper motor position for resonance at the current VFO frequency.
   * Handles user interactions, such as controlling the **Push-to-Talk (PTT)** button, which is also displayed on the TFT.   
     xx for testing SWR once stepper reached tehortical target position…..
2. **ESP32 Slave (Access Point - AP)**:
   * Responsible for controlling the **stepper motor** that adjusts the resonance of the MLA.
   * Stores a detailed **lookup table** in its **flash memory** that correlates stepper motor positions to resonance frequencies. This table is used to calculate the stepper motor position for a given frequency.  
     xx or current resonance frequency based on stepper position (used when we start and retrieved current stepper pos from prefereneces…)
   * Communicates with the **ESP Master** over a **2.4 GHz ad-hoc network**.

Xx and also with the PC via same network during characterization phase….

* + Maintains the **last stepper motor position** in its preferences, so it knows the current stepper position at startup.

1. **Icom IC 705 Transceiver**:
   * Communicates with the **ESP32 Master** via Bluetooth to provide the current **VFO frequency** and allows control of the **PTT**.  
     xx and also changing mode, i.e. switvhing from current mode to FM for SWR test and tehn switching back after the test to “Current mode”
2. **NanoVNA**:
   * Used during the **characterization phase** to measure resonance frequencies across various stepper motor positions, helping to build the lookup table that the **ESP32 Slave** stores and uses during operation.
3. **PC Configuration App**:
   * A **Python program** running on a PC that interacts with both the **NanoVNA** and the **ESP32 Slave** during the **characterization phase**.
   * The PC is used to build a detailed lookup table by measuring the resonance frequencies at different stepper positions and interpolating between these points to create a fine-grained map of 5000 frequencies and positions.
   * After the characterization is completed, the lookup table is uploaded to the **ESP32 Slave**, and the PC is no longer required for day-to-day operation.

**Characterization Phase (PC Involvement):**

Before the system can operate independently, a **characterization phase** is conducted using a **PC** connected to both the **NanoVNA** and the **ESP32 Slave**. Here’s how it works:

1. **Measuring Resonance Frequencies**:
   * The **Python program** communicates with the **NanoVNA** to measure resonance frequencies at various **stepper motor positions**.
   * Small **delta steps** are made to accurately map the antenna's resonance characteristics across a specified frequency range (e.g., from the start of a ham band to the end of that band).  
     xx lets use a concrete example saying e.g. from 14 to 14.3 MHz for th e20 meter band for instance.
2. **Building the Lookup Table**:
   * The measured data is used to build a **lookup table** that correlates each **stepper motor position** with a specific **resonance frequency**.
   * This lookup table is then **interpolated** to fill in the gaps, creating a detailed table with **5000 frequency/stepper position pairs**.
3. **Storing the Lookup Table**:
   * Once the lookup table is complete, it is uploaded to the **ESP32 Slave's flash memory**. This allows the **Slave** to retain the table across power cycles.
   * After this phase, the PC is no longer required for regular operation, and the system can function autonomously.

**Operational Phase (Without PC):**

Once the **lookup table** is stored in the **ESP32 Slave**, the system can operate independently, adjusting the **MLA** to match the VFO frequency from the **IC 705**. Here's how the process works:

1. **System Startup**:
   * When the system powers on, the **ESP32 Slave** retrieves the **last known stepper motor position** from its preferences (non-volatile memory).
   * The **ESP32 Master** sends a request to the **ESP32 Slave** for the **current antenna status**. The **Slave** responds with the current **stepper motor position** and the **resonance frequency** based on this position.
   * This information is displayed on the **TFT screen** connected to the **ESP Master**, allowing the user to see the current state of the antenna.
2. **VFO Frequency Changes**:
   * When the user turns the **VFO knob** on the **Icom IC 705**, the **ESP32 Master** detects the change in frequency via Bluetooth.
   * The **new VFO frequency** is displayed on the **TFT screen**.
   * The **ESP32 Master** then sends a request to the **ESP32 Slave** asking for the **stepper motor position** corresponding to the new VFO frequency.
3. **Querying the ESP Slave for Stepper Position**:
   * The **ESP32 Slave** uses the **lookup table** stored in its **flash memory** to calculate the **stepper position** for the new VFO frequency.
   * It also calculates the **delta steps** (i.e., how many steps the motor needs to move to reach the new resonance position from the current position).
   * The **ESP32 Slave** then sends the **new stepper position** and the **delta steps** back to the **ESP32 Master**, which displays this information on the TFT screen.
4. **Tuning the Antenna**:
   * If the user presses the **PTT button** twice in quick succession on the **Icom IC 705 microphone**, the **ESP32 Master** detects this and sends a command to the **ESP32 Slave** to move the stepper motor by the required number of **delta steps**.
   * This automatically tunes the **MLA** to the correct resonance frequency, based on the current VFO frequency from the transceiver.

**Key Features:**

* **Autonomous Tuning**: Once the system is set up, it can autonomously adjust the MLA's resonance frequency to match the VFO frequency from the IC 705, without the need for a PC.
* **Bluetooth Control**: The **ESP32 Master** retrieves the VFO frequency from the IC 705 via Bluetooth and allows the user to control the PTT button directly from the TFT interface.  
  xx as mentioned, the aim is not to use the virtual PTT button on the TFT for trasnsceiver operation in a QSO, but just for keying during 3 to 5 seconds and display the SWR retrieved via bluetooht (CAT control….)
* **Lookup Table in Flash Memory**: The detailed **5000-entry lookup table** is stored in the **ESP32 Slave’s flash memory**, allowing for precise stepper motor control based on the desired frequency.
* **Persistent Stepper Position**: The **ESP32 Slave** stores the **last stepper position** in its preferences, allowing the system to retain its state even after a power cycle.
* **Real-time Display**: The **ESP32 Master** constantly updates the TFT screen with the current VFO frequency, the calculated stepper motor position, and the steps required to reach the correct resonance.

**Conclusion:**

This system offers a complete solution for automating the tuning of a Magnetic Loop Antenna (MLA) to match the VFO frequency of an **Icom IC 705 transceiver**. After the initial characterization phase, where the resonance frequencies and stepper motor positions are mapped and stored, the system operates independently, adjusting the antenna's resonance without requiring a PC. The use of an **ESP32 Master-Slave setup**, along with a **Bluetooth link** to the transceiver, provides real-time feedback and control, ensuring seamless operation for the user.