Chris Bake

- Web Based VIC
  - 1-10 Channel Analog Amplitude Modulator
  - ESP32 Rest API, Pulse Clock & VIC Configurator
- Phase Selectable VIC
- CrayonXA (My First 9XA)
- Crossover Voltage Switch
- 8XA Alternator Setup
- DIY EMF Detector
- Magnetic Amplifiers
  - Saturation Oscillator
  - Magamp LFO standing wave oscillator
  - Introducing harmonics into Amplitude Modulation
- Bubbles
  - Everything Vibrating in a WFC
  - Demonstrating High Voltage present at the WFC
- 4 Channel 12AX7 Toroidal Amplitude Modulator
- 12AX7 Dual Triode operating in Heptode Mode
- 50W PushPull Tube Amp for driving a VIC
- Arduino / ESP32 Projects
  - ESP32 - Complex Waveform Generator V2
  - ESP32 - CWG Driver & Amplification
  - ESP32 - Complex Waveform Generator V3
  - 2 Arduino - Dual Channel - Triple AND Gate (Perfect Pulse Driver)
- Core Permeability

Web Based VIC

Web Based VIC

1-10 Channel Analog Amplitude Modulator

265079631_3195161367363934_1991161274925674298_n.png

Image not found or type unknown

WFC-Waveform-Generator (1).png

Image not found or type unknown

Web Based VIC

ESP32 Rest API, Pulse Clock & VIC Configurator

Phase Selectable VIC

Utilizing DPDT toggle switches to allow inline phase inversion of each VIC component allow significant selectivity in operational configurations.

450VDC push-pull EL34 vacuum tube driven VIC 4:1 stepdown ferromagnetic core

https://www.youtube.com/embed/YI8Mc8Q-tLQ

Earlier version of above, but with an added DPDT relay driven by a 3rd signal input and darlington driver, attempting the Crossover Voltage Switching utilized in the Steam Resonator.

https://www.youtube.com/embed/xCN0GVv7B1A

With the monster transformer in there, I would get a very strange buildup of electrostatic vibrations on the metal chassis. No vibrations were felt on the plastic of the meter panels, but everywhere on the metal parts.

https://www.youtube.com/embed/aYtpPiAnPqI

CrayonXA (My First 9XA)

Learning transistors, power supplies, 555 timers, Schmidt Triggers, AND gates, darlington transistors and optocouplers all in one shot. It still amazes me this ever worked :)

The circuit for this design originated from Max Miller's 9XA (drawn in crayon)

240662288_4705169929505824_3723763710195030686_n.jpg Image not found or type unknown	255806338_4705169809505836_119894751386171394_n.jp Image not found or type unknown
77026487_2720040918018745_400836674826797056_n (1).jpg Image not found or type unknown	

image-1656200942895.49.00.png

Max Millers' 9XA Image not found or type unknown

A funny video illustrating how I still had much to learn about building circuits :)

https://www.youtube.com/embed/2tQNh7vZGUA?t=7s

More progress on the 9XA. Learning how proper transistor biasing and loading is important for controlling amplitude modulation.

https://www.youtube.com/embed/GF0GO3bbr0s

Crossover Voltage Switch

109308463_3285262641496567_1211971226813383673_n.jpg

Image not found or type unknown

8XA Alternator Setup

This was an old attempt at using an unrectified alternator's 3 phase output running from a 1/3 HP AC synchronous motor.

https://www.youtube.com/embed/8RBypEYAaBE

DIY EMF Detector

Easy homemade EMF probe. I added a piezo buzzer to mine, and found it very helpful to "listen" to the fields coming off of things like wires and HV bulbs, and water capacitors.

DIY Guide: https://www.instructables.com/Make-a-Ghost-Detector/

F2VE0JKJ153YCTJ.webp

Image not found or type unknown

https://www.youtube.com/embed/p_x_Ax6RarM

Magnetic Amplifiers

Saturation Oscillator

> You must *listen* to your energy. It plays notes. Once you learn to trigger those notes, you can play a chord.

Playing with distributed inductances and capacitances and harmonics generated from 60 Hz line noise. Using resistive dampening to load mutually coupled transformers inline with the magnetic amplifier's reactive side.

https://www.youtube.com/embed/3_nBHoaIbeU

Magnetic Amplifiers

Magamp LFO standing wave oscillator

https://www.youtube.com/embed/J0oaSOzNXkQ?t=816s

Magnetic Amplifiers

Introducing harmonics into Amplitude Modulation

https://www.youtube.com/embed/-vXpCsLjIgE

Bubbles

Bubbles

Everything Vibrating in a WFC

This shows Max's Double Barrel cell holder adapted to mount electrodes at opposing ends of each tube.

Tiny streams of nanobubbles can be seen coming from the bolt threads

https://www.youtube.com/embed/0ImUI5C9VYA

Bubbles

Demonstrating High Voltage present at the WFC

https://www.youtube.com/embed/rP-eQOGTpL8

"Waterless" energy transfer

https://www.youtube.com/embed/E-08meTh-8Y

Visualizing standing waves occurring across the WFC capacitor.

https://www.youtube.com/embed/y7tQPvy88I4

4 Channel 12AX7 Toroidal Amplitude Modulator

Early Build pics of the 4 Channel modulator.

133798005_3767217696634390_7095165259752784715_n.jpg Image not found or type unknown	134047555_3767217633301063_1100532127399627698_n.j Image not found or type unknown
133600591_3767218093301017_8668741515042871589_n.jpg Image not found or type unknown	133708014_3767218326634327_3937906867011726984_n.j Image not found or type unknown

12AX7 Dual Triode operating in Heptode Mode

Truly Analog Amplitude Modulation AND Gating

125466214_3650706548285506_994606898162063674_n.jpg

Image not found or type unknown

50W PushPull Tube Amp for driving a VIC

74176231_2666543493368488_4586020633896288256_n.jpg

Image not found or type unknown

73019663_2666543533368484_3939437153141915648_n.jpg

Image not found or type unknown

Arduino / ESP32 Projects

Arduino / ESP32 Projects

ESP32 - Complex Waveform Generator V2

Setting Up The App

> ESP32 Complex Waveform Generator - Arrangement for WROOM-32 or WROVER-E (DevKit-C)

https://www.youtube.com/embed/AxDYTR773SM

PARTS Required

- 1 - ESP32 (WROOM-32 or WROVER-32) with 16 exposed pins
- 1 - ESP32 Breakout Board or equivalent pin header block
- 7 - Rotary Encoders (i.e. KY-040 Rotary Encoder Module CYT1062)
  - TODO: add >= 2 more for Elongation adjustments.
- 1 - +5VREG 1A Power Supply for ESP32 (i.e. ATX PowerSupply)

Installation Prerequisites

Install ESP Libraries in Arduino-IDE v2.0

```
ArduinoJson
ESP32Encoder
```

Step 1. Open Esp32Full.ino and set Wifi Credentials

```
// #### Change Me - Local Wifi Info ####
const char *SSID = "NETGEAR";
const char *PWD = "12345678";
```

Step 2. Configure free local LAN IP address

Check your Router for more information

```
// #### CHANGE ME ####
// Set your Static IP address to a free IP in your local network
IPAddress local_IP(192, 168, 1, 8);
// Set your Gateway IP address
IPAddress gateway(192, 168, 1, 1);

IPAddress subnet(255, 255, 255, 0);
IPAddress primaryDNS(8, 8, 8, 8);   //optional
IPAddress secondaryDNS(8, 8, 4, 4); //optional
```

~~Step 3. Configure ESP_HOST in Javascript File~~

~~Edit `./assets/espwavegen.js` and set the IP address used in Step 2 above.~~

> ~~TODO: Make configurable in the web interface~~

~~Save and close the file~~

Step 3. Upload The code in `Esp32Full.ino`

Paste the code into your Arduino-IDE and upload it to your ESP32

> Installing the ESP32 Board in Arduino IDE

Step 4. Access the WebApp in your Web Browser

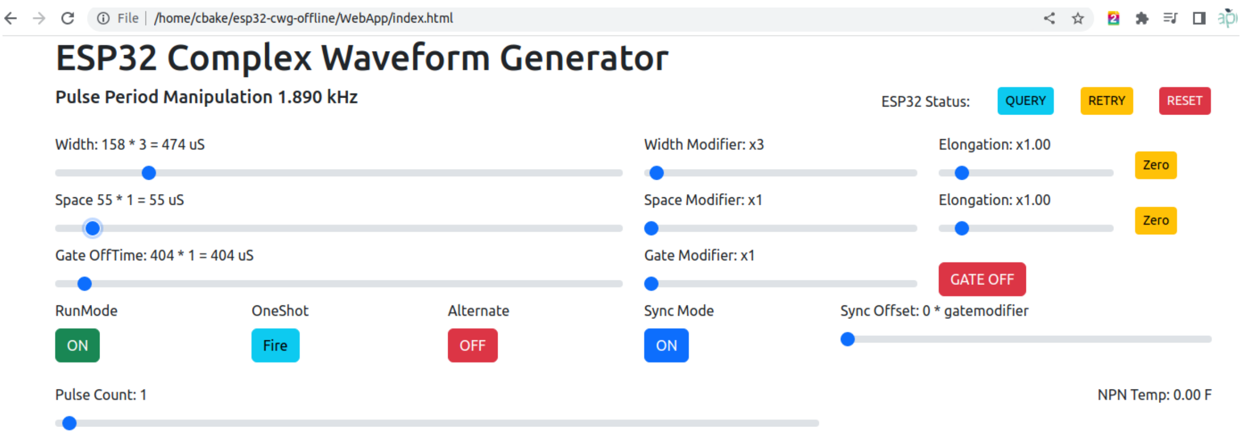
Open Web Browser
Open `index.html` in the `WebApp` directory below this file

- File -> Open -> Browse to WebApp/index.html -> Open

Interface is now displayed!

Step 5: Enjoy!!

Please post pics and videos of your waves, and let others know how achievable this is!



File | /home/cbake/esp32-cwg-offline/WebApp/index.html

ESP32 Complex Waveform Generator

Pulse Period Manipulation 1.890 kHz

ESP32 Status: QUERY RETRY RESET

Width: 158 * 3 = 474 uS

Width Modifier: x3

Elongation: x1.00 Zero

Space 55 * 1 = 55 uS

Space Modifier: x1

Elongation: x1.00 Zero

Gate OffTime: 404 * 1 = 404 uS

Gate Modifier: x1

GATE OFF

RunMode: ON

OneShot: Fire

Alternate: OFF

Sync Mode: ON

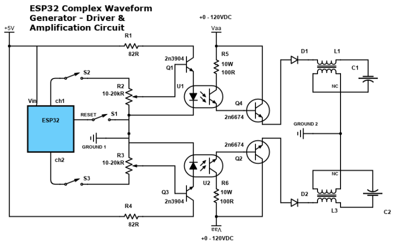
Sync Offset: 0 * gatemodifier

Pulse Count: 1

NPN Temp: 0.00 F

Need Amplification?

See: ESP32 - Complex Waveform Generator - Driver & Amplification



Additional Troubleshooting / Customization

Optional: Configure alternate Output Pins

- Output will be on Pins D2 and D4 by default

```
// #### Output Pins ####
int pinChannel1 = 2;
int pinChannel2 = 4;
```

Optional: Adjust Encoder Pins if needed

```
int pulseCount_EncoderPIN1 = 14;
int pulseCount_EncoderPIN2 = 13;
int pulseWidth_EncoderPIN1 = 35;
int pulseWidth_EncoderPIN2 = 34;
int pulseSpace_EncoderPIN1 = 19;
int pulseSpace_EncoderPIN2 = 18;
int gate_freq_EncoderPIN1 = 22;
int gate_freq_EncoderPIN2 = 23;

int pulseWidthModifier_EncoderPIN1 = 27;
int pulseWidthModifier_EncoderPIN2 = 26;
int pulseSpaceModifier_EncoderPIN1 = 5;
int pulseSpaceModifier_EncoderPIN2 = 32;
int gateModifier_EncoderPIN1 = 25;
int gateModifier_EncoderPIN2 = 33;
```

ESP C+ Code

Git Repo: https://bitbucket.org/cbake6807/esp32-complex-waveform-generator/src/master/

Troubleshooting:

View the Console Log for errors in your browser while clicking the app's sliders buttons etc..

Look in the Network tab for red errors. 404 or other. Sometimes the ESP may drop or reject the connection on the first attempt. Just refresh the browser once or twice and it should resolve.

Screenshot from 2022-08-06 13-24-55.png

Image not found or type unknown

> https://www.browserstack.com/guide/inspect-element-in-chrome#:~:text=One%20of%20the%20easiest%20ways,%2C%20Sources%2C%20and%20other%20tools.

Confirm the ESP is a connected host in your network and was given the IP you specified

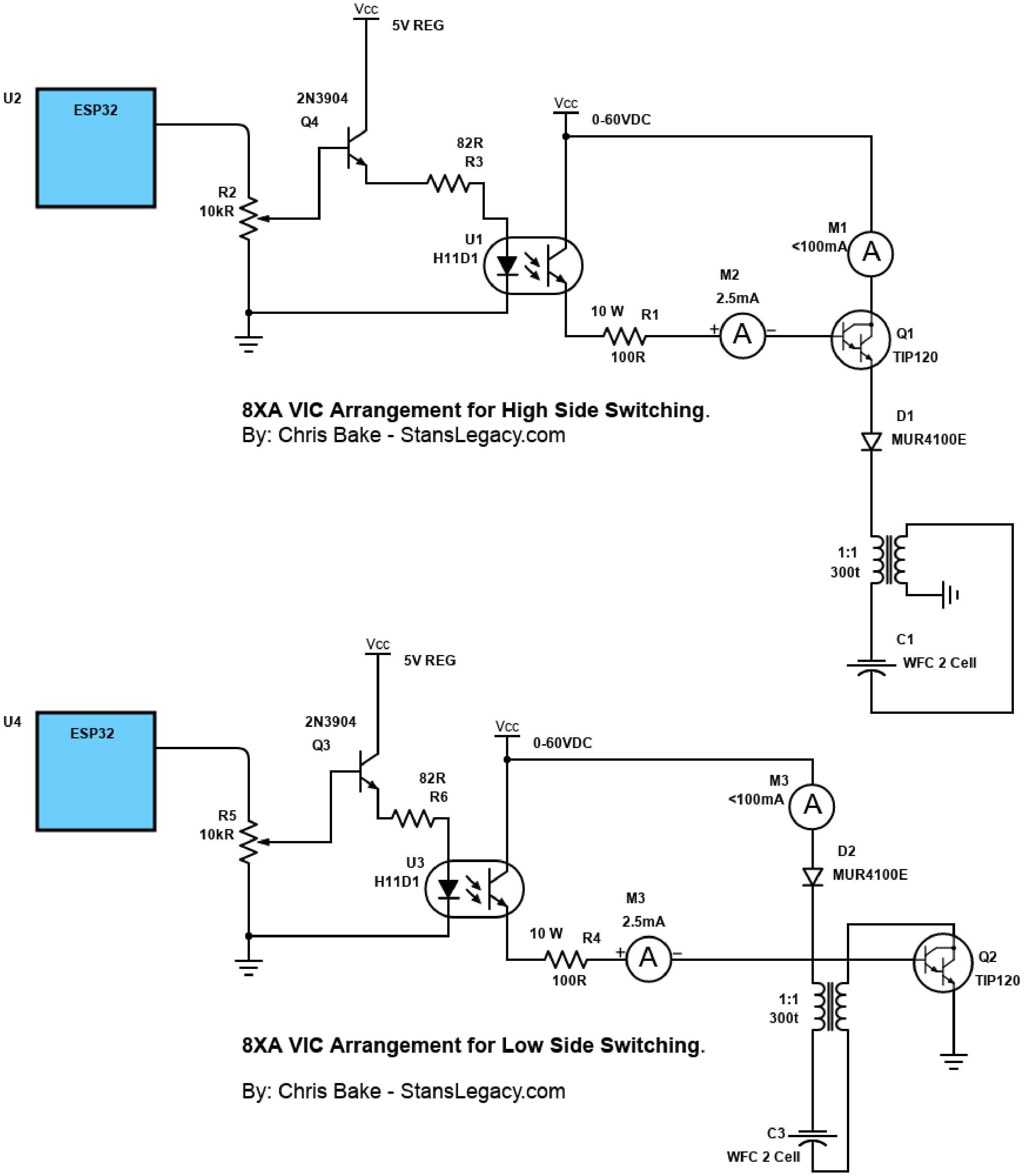
> https://www.wikihow.com/See-Who-Is-Connected-to-Your-Wireless-Network

Code notes if internet connectivity isn't an option. Also, encoder wiring connections for this particular setup (Notepad++ document file type). Chris Bake ESP32 notes.txt

Notepad++ download: Notepad++

ESP32 - CWG Driver & Amplification

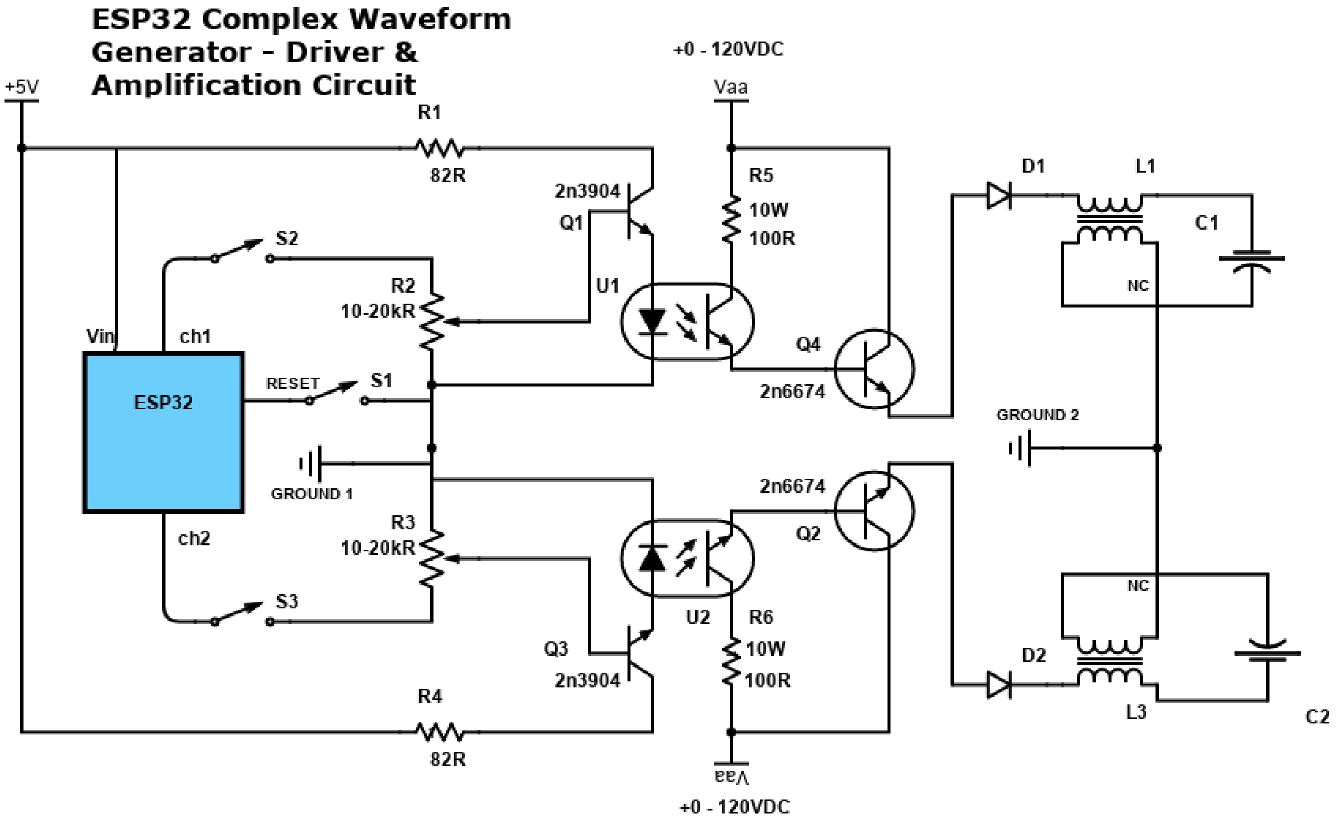
A dual channel, dual isolated power supply, pulse amplification for driving all VIC arrangements



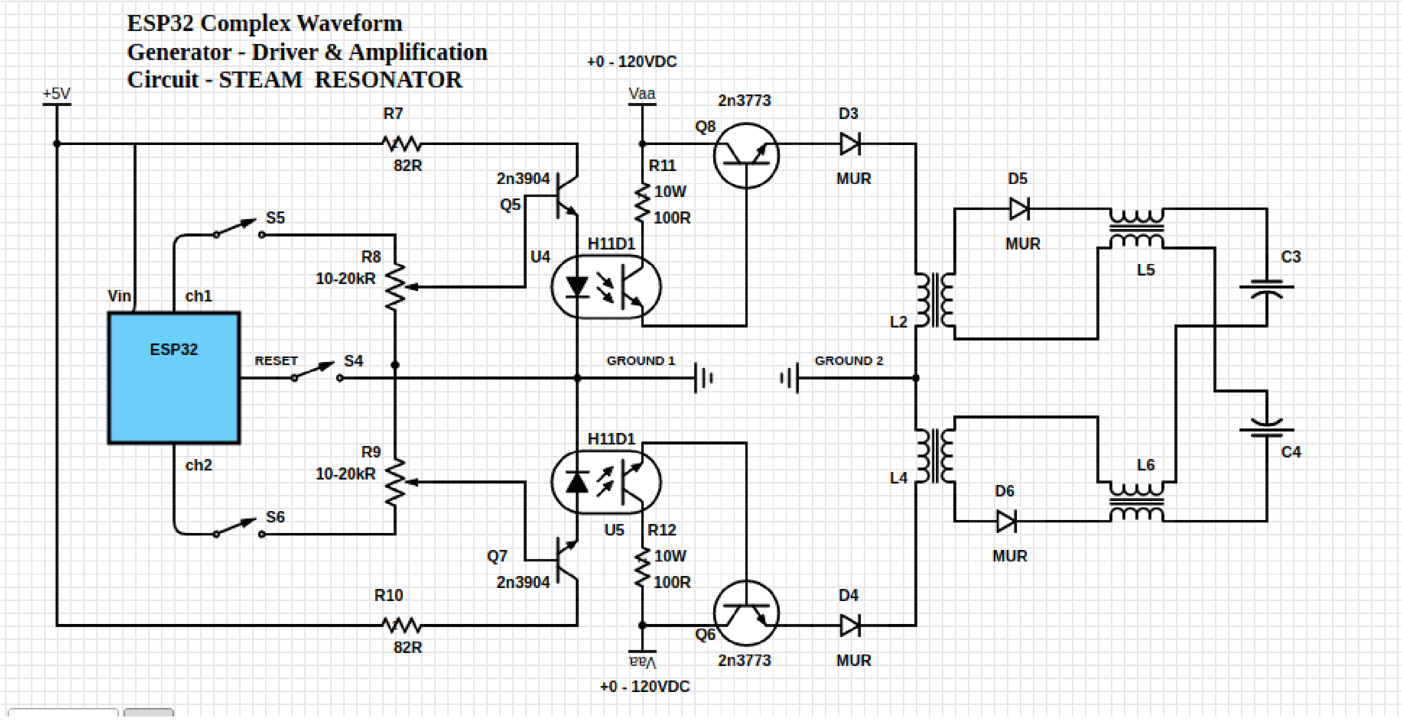
8XA VIC Arrangement for High Side Switching.
By: Chris Bake - StansLegacy.com

8XA VIC Arrangement for Low Side Switching.

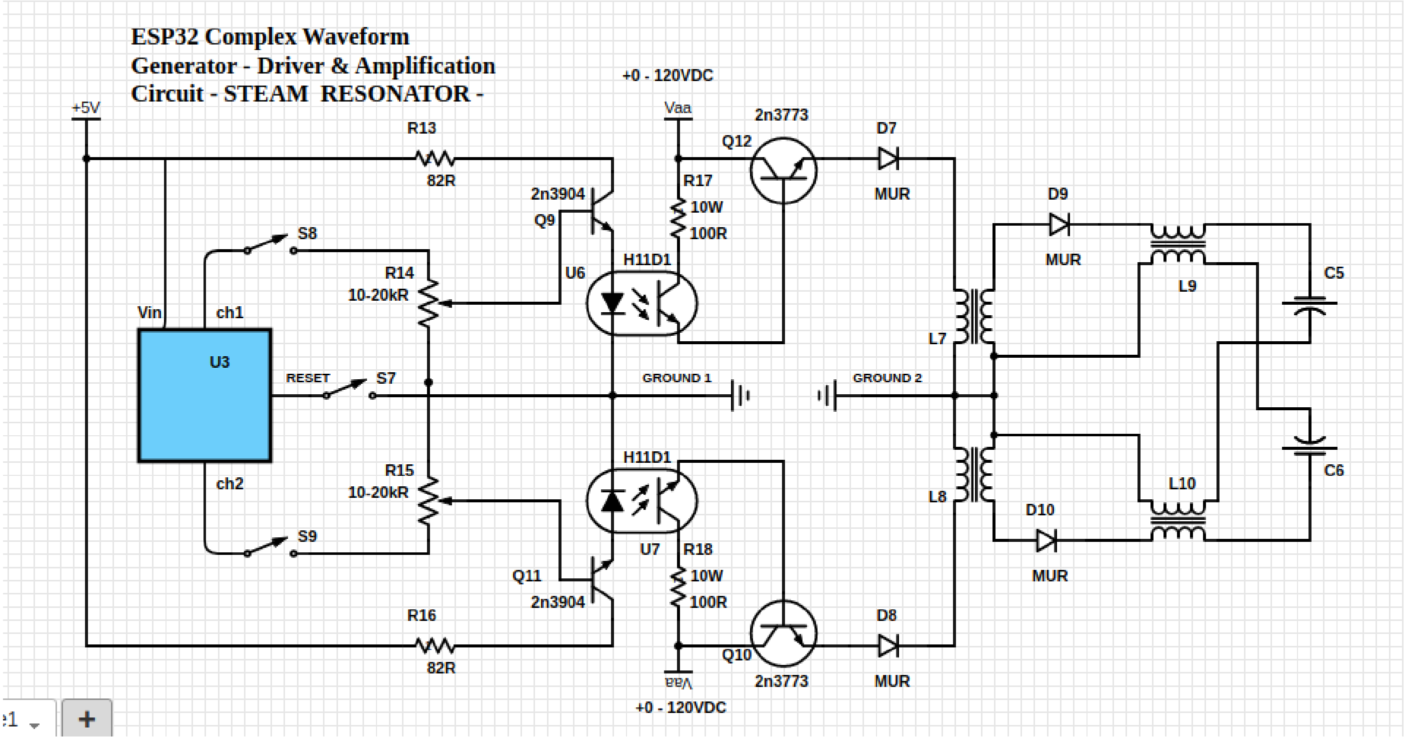
By: Chris Bake - StansLegacy.com



Arrangement favoring Particle Oscillation as an Energy Generator



Variation 2 - Ground Bonded / "Forced" Uni-polar Operation



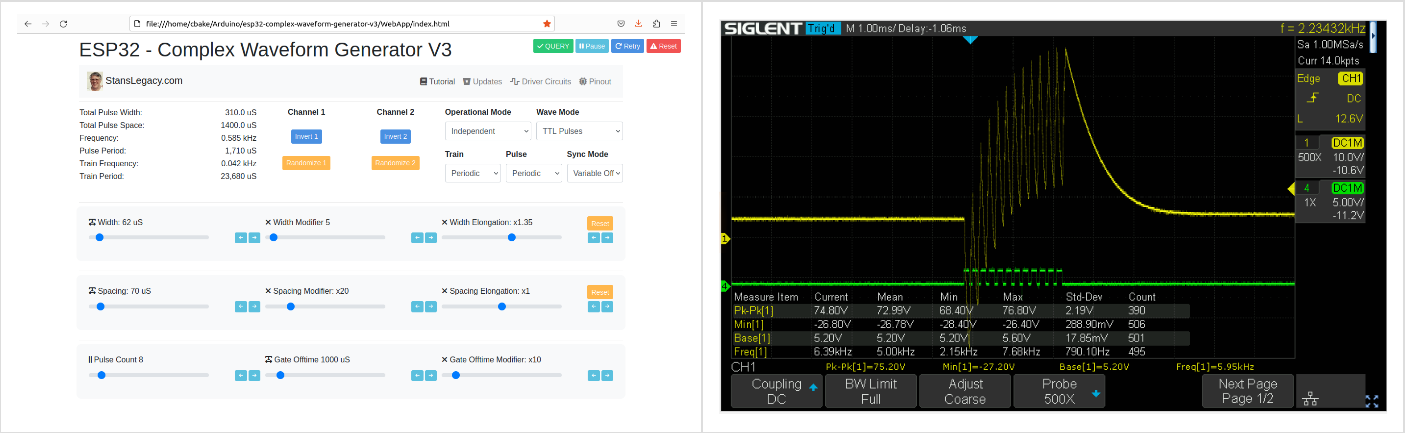
ESP32 Complex Waveform Generator - Driver & Amplification Circuit - STEAM RESONATOR -

Arduino / ESP32 Projects

ESP32 - Complex Waveform Generator V3

Installing and Using the ESP32 Complex Waveform Generator V3 Application

Screenshot of web interface



file:///home/cbake/Arduino/esp32-complex-waveform-generator-v3/WebApp/index.html

ESP32 - Complex Waveform Generator V3

Total Pulse Width: 310.0 uS
Total Pulse Space: 1400.0 uS
Frequency: 0.585 kHz
Pulse Period: 1,710 uS
Train Frequency: 0.042 kHz
Train Period: 23,680 uS

Operational Mode: Independent
Wave Mode: TTL Pulses
Train: Periodic
Pulse: Periodic
Sync Mode: Variable Off

Width: 62 uS
× Width Modifier 5
× Width Elongation: x1.35
Spacing: 70 uS
× Spacing Modifier: x20
× Spacing Elongation: x1
Pulse Count 8
Gate Offtime 1000 uS
× Gate Offtime Modifier: x10

SIGLENT Trig'd M 1.00ms/ Delay:-1.06ms f = 2.23432kHz

Sa 1.00MSa/s
Curr 14.0kpts
Edge CH1 DC
L 12.6V
1 DC1M 500X 10.0V/ -10.6V
4 DC1M 1X 5.00V/ -11.2V

Measure Item	Current	Mean	Min	Max	Std-Dev	Count
Pk-Pk[1]	74.80V	72.99V	68.40V	76.80V	2.19V	390
Min[1]	-26.80V	-26.78V	-28.40V	-26.40V	288.90mV	506
Base[1]	5.20V	5.20V	5.20V	5.60V	17.85mV	501
Freq[1]	6.39kHz	5.00kHz	2.15kHz	7.68kHz	790.10Hz	495

CH1 Pk-Pk[1]=75.20V Min[1]=-27.20V Base[1]=5.20V Freq[1]=5.95kHz

Coupling: DC
BW Limit: Full
Adjust: Coarse
Probe: 500X

Prerequisites

To use this application, you need to have the Arduino IDE installed on your computer. You can download the Arduino IDE from the official website: https://www.arduino.cc/en/software

1. **ESP32 Development Board**: You'll need an ESP32 development board, such as the popular ESP32-DevKitC or ESP32-WROOM-32D. These boards typically come with Wi-Fi and Bluetooth capabilities and a variety of GPIO pins for interfacing with peripherals.
2. **Rotary Encoders**: To adjust the waveform parameters, you will need a total of 7 rotary encoders. You can use KY-040 rotary encoder modules or any other type of incremental rotary encoder with built-in push buttons. Ensure that the rotary encoders you choose have a CLK, DT, and SW (push button) pinout.
3. **Breadboard and Jumper Wires**: A breadboard and jumper wires are required to make the necessary connections between the ESP32 development board and the rotary encoders.

4. **Power Supply**: You will need a power supply to power the ESP32 development board. This can be a USB power supply, a battery, or any other suitable power source that meets the board's voltage and current requirements, such an ATX Supply from a computer. 5VDC 1A minimum is recommended to prevent brownout conditions while interacting with a driver circuit.
5. **Oscilloscope (optional)**: To visualize the generated waveform, you can use an oscilloscope. Connect the output pins (channel1OutputPin and channel2OutputPin) from the ESP32 development board to the oscilloscope's input channels.

Installing required libraries

1. **ArduinoJson**: To install the ArduinoJson library, follow these steps: a. Open the Arduino IDE. b. Click on `Tools` in the menu bar, then `Manage Libraries`. c. In the Library Manager window, search for "ArduinoJson" in the search bar. d. Find "ArduinoJson by Benoit Blanchon" in the search results and click on the `Install` button.
2. **ESP32Encoder**: To install the ESP32Encoder library, follow these steps: a. Open the Arduino IDE. b. Click on `Tools` in the menu bar, then `Manage Libraries`. c. In the Library Manager window, search for "ESP32Encoder" in the search bar. d. Find "ESP32Encoder by Gil Mora" in the search results and click on the `Install` button.

Uploading the Application

1. Download the source code for the ESP32 Complex Waveform Generator V3 application or copy it to a new file in the Arduino IDE.
2. Connect the ESP32 development board to your computer using a USB cable.
3. In the Arduino IDE, select the appropriate board and port under `Tools` > `Board` and `Tools` > `Port`.
4. Click on the `Upload` button (right-facing arrow icon) in the Arduino IDE to compile and upload the application to the ESP32 development board.

Hardware Setup

1. Wire the rotary encoders and other components according to the pin assignments defined in the source code.

ESP32 Pin	Encoder Connection	Encoder Function
14	Pulse Encoder CLK	Pulse count
13	Pulse Encoder DT	Pulse count
35	Width Encoder CLK	Pulse width
34	Width Encoder DT	Pulse width

ESP32 Pin	Encoder Connection	Encoder Function
19	Spacing Encoder CLK	Pulse spacing
18	Spacing Encoder DT	Pulse spacing
23	Off-time Encoder CLK	Off-time
22	Off-time Encoder DT	Off-time
27	Width Mod Encoder CLK	Width modifier
26	Width Mod Encoder DT	Width modifier
15	Spacing Mod Encoder CLK	Spacing modifier
32	Spacing Mod Encoder DT	Spacing modifier
33	Off-time Mod Encoder CLK	Off-time modifier
4	Off-time Mod Encoder DT	Off-time modifier
2	Output Channel 1	Output waveform
5	Output Channel 2	Output waveform

2. Make sure the connections are secure and verify the CLK/DT pins for each encoder are wired correctly, and that each encoder turns in the correct direction relative to the respective parameter change.

Using the Application

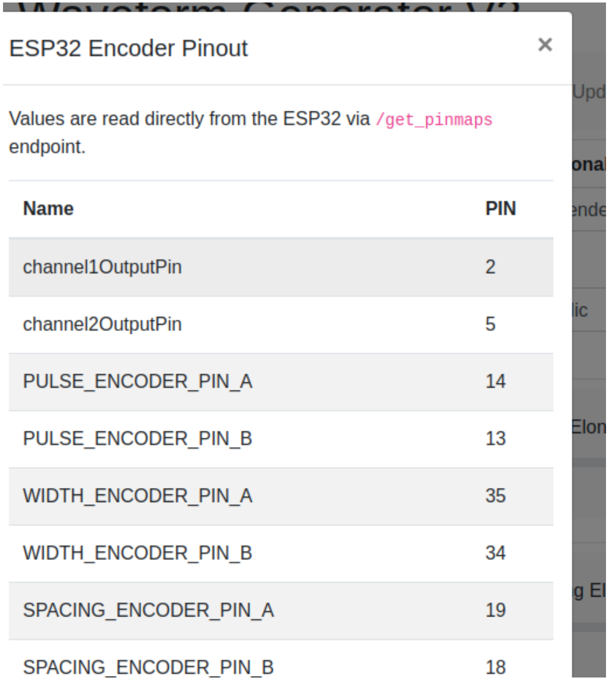
1. Power on the ESP32 development board.
2. Use the rotary encoders to adjust the following parameters:
   - Pulse count
   - Pulse width
   - Pulse spacing
   - Off-time
   - Width modifier
   - Spacing modifier
   - Off-time modifier
3. The application will generate a complex waveform based on the adjusted parameters.
4. Connect the output pins (channel1OutputPin and channel2OutputPin) to an oscilloscope to visualize the generated waveform.
5. Open the WebApp/index.html page in a browser.
6. Fine-tune the parameters using the rotary encoders to achieve the desired waveform shape and characteristics.

Next Steps For Utilization

- Build ESP32 CWG - VIC Driver circuit(s) - https://stanslegacy.com/books/chris-bake/page/esp32-cwg-driver-amplification
- Cell Construction - https://stanslegacy.com/books/ethan-crowder/page/resonant-cavity-related

Troubleshooting

1. If the waveform does not match the expected output, verify the wiring connections and ensure the rotary encoders are functioning correctly.
2. If the application does not upload to the ESP32 development board, double-check the board and port selection in the Arduino IDE.
3. If the rotary encoders behave unexpectedly (e.g., adjusting one parameter affects another), check the CLK/DT pin assignments and wiring.
4. Click the PInout button to fetch the currently defined GPIO pins from the ESP32 directly and confirm they are wired correctly.



ESP32 Encoder Pinout

Values are read directly from the ESP32 via `/get_pinmaps` endpoint.

Name	PIN
channel1OutputPin	2
channel2OutputPin	5
PULSE_ENCODER_PIN_A	14
PULSE_ENCODER_PIN_B	13
WIDTH_ENCODER_PIN_A	35
WIDTH_ENCODER_PIN_B	34
SPACING_ENCODER_PIN_A	19
SPACING_ENCODER_PIN_B	18

For further assistance or to report any issues, contact the application's support team or refer to the community forums.

2 Arduino - Dual Channel - Triple AND Gate (Perfect Pulse Driver)

By: Chris Bake

Arduino Code: https://bitbucket.org/cbake6807/dualtripleseq/src/master/

Parts List

1. **External Signal Generator**: 0-5Vppk output.
2. **Power Supply**: ATX is ideal, providing +5V REG and +12V REG.
3. **2N7000 Signal MOSFETs**: Quantity 6.
4. **IRFP460 or Similar Power N-channel MOSFET**: Quantity 2.
5. **56Ω 1/8W Resistors**: Quantity 10.
6. **100Ω 1/2W Resistors**: Quantity 2.
7. **4.7kΩ Resistors**: Quantity 2.
8. **2N3906 PNP General Purpose Transistor**: Quantity 2 (can be substituted with any general PNP transistor).
9. **IR2110PB Gate Driver Chip 14-pin**: Quantity 2.
10. **Arduino Nano (or similar)**: Quantity 2 (must support hardware PCNT).
11. **Rotary Encoder**: Quantity 1.

Software Requirements

- **Arduino IDE**: Ensure it is installed and updated to the latest version.
- **Encoder Library**: Install via the Arduino Library Manager.

Arduino Setup

Pulse Counter Arduino

1. **Upload Script**
   - Open the Arduino IDE.
   - Connect the first Arduino (PulseCounter) to your PC.
   - Open `PulseCounter.ino` from the provided file.
   - Upload the script to the Arduino.
2. **Connect the Encoder**
   - Connect the encoder's VCC to the Arduino's 5V pin.
   - Connect the encoder's GND to the Arduino's GND pin.
   - Connect the encoder's CLK and DT pins to two digital pins on the Arduino D2 and D3.
   - Connect the encoder's Button pin to D5.
3. **Verify Encoder Output**
   - Open the Serial Monitor in the Arduino IDE.
   - Rotate the encoder and check the output to confirm it is functioning correctly.

Adding a Pushbutton for Sync Mode Toggle

To add a pushbutton to the Sequencer Arduino for toggling the sync mode, follow these instructions:

Parts Required

- **Pushbutton**: Quantity 1
- **10kΩ Resistor**: Quantity 1
- **Connecting Wires**

Hardware Connections

1. **Connect the Pushbutton**
   - Connect one terminal of the pushbutton to the Arduino's 5V pin.
   - Connect the other terminal of the pushbutton to digital pin 4 on the Arduino.
2. **Add a Pull-Down Resistor**
   - Connect a 10kΩ resistor between digital pin 4 and GND. This ensures that the pin reads LOW when the button is not pressed.

Summary of Connections

- **Pushbutton Terminal 1** → Arduino 5V
- **Pushbutton Terminal 2** → Arduino Pin 4
- **10kΩ Resistor** → Arduino Pin 4 and GND

Sequencer Arduino

1. **Upload Script**

- Disconnect the PulseCounter Arduino and connect the second Arduino (Sequencer) to your PC.
- Open `Sequencer.ino` from the provided file.
- Upload the script to the Arduino.

Pin Mapping and Connections

Connecting the Two Arduinos

- **PulseCounter Arduino to Sequencer Arduino**
  - PulseCounter D5 (Input) ← Signal Generator
  - PulseCounter D9 (Output) → Sequencer D2 (Input)
- **Sequencer Arduino Outputs**
  - Sequencer D9 (Output) → Input to the first AND gate tree
  - Sequencer D10 (Output) → Input to the second AND gate tree

Note: The schematic shows the outputs merged due to limitations, but there should be dual outputs from the sequencer, each connecting to a separate AND gate tree.

- **Shared Connections**
  - Both Arduinos' GND pins should be connected together to ensure a common ground.

Gate Driver Chip Connections

IR2110PB Gate Driver Chip

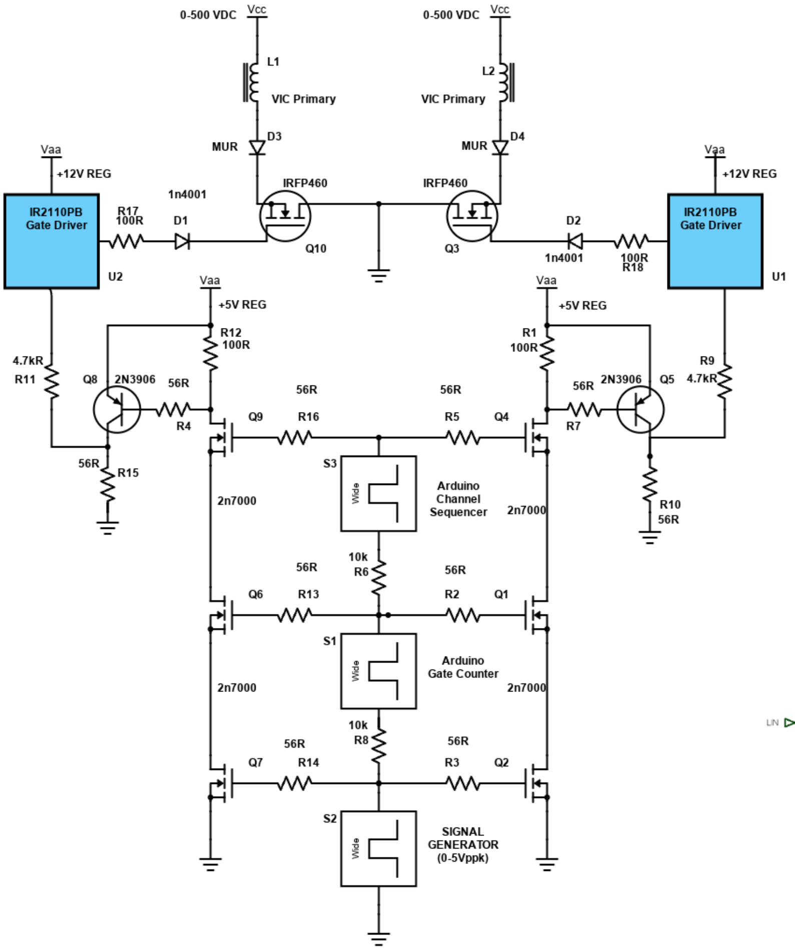
1. **Power Connections**
   - VCC (Pin 3) → +12V REG
   - VSS (Pin 12) → GND
2. **Input Connections**
   - LIN (Pin 11) → Arduino PWM pin (as per script)
   - HIN (Pin 10) → Arduino PWM pin (as per script)
3. **Output Connections**
   - LO (Pin 1) → Gate of the IRFP460 MOSFET
   - HO (Pin 7) → Gate of the second IRFP460 MOSFET
   - VS (Pin 6) → Source of the high-side MOSFET
4. **Bootstrap Capacitor**
   - Connect a 0.1µF capacitor between VB (Pin 8) and VS (Pin 6).
5. **Other Components**
   - D2: Place a 1N4001 diode between VB (anode) and VCC (cathode).
   - R2: Connect a 10Ω resistor between LO and the gate of the MOSFET.

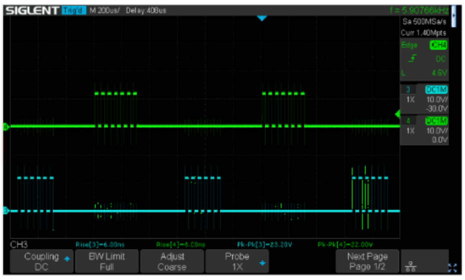
Schematic Overview

Refer to the schematic image to visualize these connections. The IR2110PB gate driver chips control the IRFP460 MOSFETs, enabling high-power switching of the VIC (Voltage Intensifier Circuit) primaries.

Additional Notes

- **Resistor Tolerances**: The resistors can have a large tolerance. The 56Ω value is selected to preserve signal clarity. Any value ≤220Ω should be acceptable.
- **Merged Outputs**: The Arduino Channel Sequencer has 2 outputs shown merged due to Scheme-It limitations.

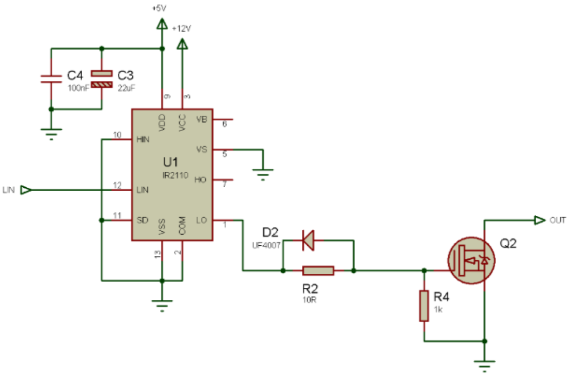




Perfect Pulse - Triple AND Gate Sequencer - Dual Gate Driver

By: Chris Bake
stanslegacy.com

Arduino code required. See Gate Driver below for additional chip connections.



Core Permeability

> There are 2 inductors on a ferrite core, closed loop U shape. The core's permeability is approx 2000. One coil is 300 turns of 22Ga wire, and the other coil is 1000 turns of 30Ga wire. The inductance of the 300 turn coil, while shorting the 1000 turn coil, is 12.188mH. The inductance of the 1000 turn coil, while shorting the 300 turn coil, is 159.69mH. Both measurements were taken at a 1 kHz test frequency with an LCR meter.
>
> **What is the true core permeability?**

We can use the following equation to calculate the inductance of an inductor on a closed loop ferrite core with two coils:

```
L_total = μ * N_1^2 * A / (2 * l) + μ * N_2^2 * A / (2 * l)
```

where L_total is the total inductance of the two coils, μ is the permeability of the core in henries per meter (H/m), N_1 and N_2 are the number of turns in each coil, A is the cross-sectional area of the core in square meters (m^2), and l is the length of the core in meters (m).

We are given that one coil has 300 turns of 22 gauge wire, and the other coil has 1000 turns of 30 gauge wire. We are also given that the inductance of the 300 turn coil while shorting the 1000 turn coil is 12.188 mH, and the inductance of the 1000 turn coil while shorting the 300 turn coil is 159.69 mH. Both measurements were taken at a 1 kHz test frequency using an LCR meter.

Let's start by solving for the permeability μ. We can do this by using the equation for the inductance of the 300 turn coil:

```
12.188 × 10^-3 H = μ * N_1^2 * A / (2 * l)
```

Substituting the given values for N_1, A, and l, we get:

```
12.188 × 10^-3 H = μ * (300)^2 * A / (2 * 0.1 m)
```

Simplifying, we get:

```
μ = 4 * 12.188 × 10^-3 H * 0.1 m / (300)^2 * A
```

The cross-sectional area of 22 gauge wire is approximately 0.0000006475 m^2, and the cross-sectional area of 30 gauge wire is approximately 0.0000005067 m^2. Since the two coils are wound on the same core, we can assume that they have the same cross-sectional area. Let's use the average of the two values:

```
A = (0.0000006475 m^2 + 0.0000005067 m^2) / 2 = 0.0000005771 m^2
```

Substituting this value, we get:

```
μ = 1.56 × 10^-3 H/m
```

Next, let's use the equation for the inductance of the 1000 turn coil to solve for the length of the core:

```
159.69 × 10^-3 H = μ * N_2^2 * A / (2 * l)
```

```
Substituting the values we found for μ and A, we get:
```

```
159.69 × 10^-3 H = (1.56 × 10^-3 H/m) * (1000)^2 * 0.0000005771 m^2 / (2 * l)
```

Simplifying, we get:

```
l = (1.56 × 10^-3 H/m) * (1000)^2 * 0.0000005771 m^2 / (2 * 159.69 × 10^-3 H)

l = 0.0266 m
```

Therefore, the length of the core is approximately 0.0266 m.

Finally, we can use the same equation for the inductance of the 1000 turn coil to solve for the true value of the permeability μ:

```
159.69 × 10^-3 H = μ * (1000)^2 * A
```

Substituting the value of A we found earlier, we get:

```
159.69 × 10^-3 H = μ * (1000)^2 * 0.0000005771 m^2 / (2 * 0.0266 m)
```

Simplifying, we get:

μ = 1865.5 H/m

> Therefore, the true permeability of the core is approximately **1865.5 H/m**.

To determine the number of turns of 30Ga wire required to produce the same inductance as the 300 turn 22Ga coil, we can use the following formula:

L = (μ * N^2 * A) / l

where L is the inductance in henries, μ is the permeability of the core, N is the number of turns, A is the cross-sectional area of the core, and l is the length of the core.

We know that the inductance of the 300 turn 22Ga coil is 12.188mH and the permeability of the core is 1865.5 H/m. We also know that the cross-sectional area of the core is 0.0000005771 m^2 and the length of the core is 0.0266 m.

Substituting these values, we get:

12.188 × 10^-3 H = (1865.5 H/m) * (300)^2 * 0.0000005771 m^2 / 0.0266 m

Solving for N, we get:

N ≈ **712 turns**

> Therefore, approximately 712 turns of 30Ga wire would be required to produce the same inductance as the 300 turn 22Ga coil.