# Teensy 4.0 Pulse Generator

A blue rectangular object with black trim

Description automatically generated

# A blue box with a black wire Description automatically generated

# A blue box with a circuit board and wires Description automatically generated

# Firmware:

// Teensy 4.0 Signal Generator

// For accurate Timing, set Teensy CPU to 816 MHz

// Electronics Workshop, Robin O'Reilly, 2024

#include <ArduinoJson.h>

// Timing Variables

unsigned long pulseInterval;

unsigned long interPulseDelay;

unsigned long pulseWidth;

const int instr\_overhead = 15; // Overhead to execute CPU instruction

// Lower limits for the parameters

const unsigned long minPulseInterval = 10; // Minimum 10ms

const unsigned long minInterPulseDelay = 20; // Minimum 20ns

const unsigned long minPulseWidth = 20; // Minimum 20ns

bool enable = 0; // Wait for first JSON before enabling

void setup() {

Serial.begin(115200);

pinMode(1, OUTPUT);

digitalWriteFast(1, LOW); // Initial state low (inactive)

delay(3000); // Give serial port time to open

Serial.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*Teensy 4.0 Signal Generator\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

Serial.println("> Usage: Send JSON string, for e.g {\"pulseInterval\": 100, \"interPulseDelay\": 200, \"pulseWidth\": 100}.");

Serial.println("> Interval (time between pulse pairs) is in milliseconds.");

Serial.println("> interPulseDelay: Time between the end of the first pulse and the start of the second pulse, in nanoseconds.");

Serial.println("> pulseWidth: Duration of each pulse, in nanoseconds.");

Serial.println("> Robin O'Reilly 2024");

}

void loop() {

if (Serial.available() > 0) {

String jsonString = Serial.readStringUntil('\n');

StaticJsonDocument<200> doc;

DeserializationError error = deserializeJson(doc, jsonString);

if (error) {

Serial.print(F(">ERR DeserializeJson() failed: "));

Serial.println(error.f\_str());

return;

} else {

unsigned long tempPulseInterval = doc["pulseInterval"];

unsigned long tempInterPulseDelay = doc["interPulseDelay"];

unsigned long tempPulseWidth = doc["pulseWidth"];

// Check and enforce lower bounds

bool outOfBounds = false;

if (tempPulseInterval < minPulseInterval) {

Serial.print(F(">ERR Out of Bounds: pulseInterval "));

Serial.print(tempPulseInterval);

Serial.print(F("ms is less than the minimum "));

Serial.print(minPulseInterval);

Serial.println(F("ms"));

outOfBounds = true;

}

if (tempInterPulseDelay < minInterPulseDelay) {

Serial.print(F(">ERR Out of Bounds: interPulseDelay "));

Serial.print(tempInterPulseDelay);

Serial.print(F("ns is less than the minimum "));

Serial.print(minInterPulseDelay);

Serial.println(F("ns"));

outOfBounds = true;

}

if (tempPulseWidth < minPulseWidth) {

Serial.print(F(">ERR Out of Bounds: pulseWidth "));

Serial.print(tempPulseWidth);

Serial.print(F("ns is less than the minimum "));

Serial.print(minPulseWidth);

Serial.println(F("ns"));

outOfBounds = true;

}

// Update values only if all are within bounds

if (!outOfBounds) {

pulseInterval = tempPulseInterval;

interPulseDelay = tempInterPulseDelay;

pulseWidth = tempPulseWidth;

// Send back the values to the PC for verification

Serial.print(F(">OK Parsed values - pulseInterval: "));

Serial.print(pulseInterval);

Serial.print(F(", interPulseDelay: "));

Serial.print(interPulseDelay);

Serial.print(F(", pulseWidth: "));

Serial.println(pulseWidth);

enable = 1;

}

}

}

if (enable == 1) {

pulseStart();

}

delay(pulseInterval);

}

void pulseStart() {

// Generate first pulse

digitalWriteFast(1, HIGH); // Start first pulse (active low)

delayNanoseconds(pulseWidth - instr\_overhead); // Pulse width in nanoseconds

digitalWriteFast(1, LOW); // End first pulse

// Wait for inter-pulse delay

delayNanoseconds(interPulseDelay - instr\_overhead); // Inter-pulse delay in nanoseconds

// Generate second pulse

digitalWriteFast(1, HIGH); // Start second pulse (active low)

delayNanoseconds(pulseWidth - instr\_overhead); // Pulse width in nanoseconds

digitalWriteFast(1, LOW); // End second pulse

}

# Serial Printout:

Baud Rate: 115,200

A screen shot of a computer

Description automatically generated

# Example Command:

Send the following to set interval to 100**ms**, pulse delay of 200**ns** and pulse width at 100**ns**:

{"pulseInterval": 100, "interPulseDelay": 200, "pulseWidth": 100}

# Example Response:



# Scope Output for Above Command:

A screen shot of a computer monitor

Description automatically generated

# Note on Interfacing:

No Logic conversion required as TTL-NIM converter will accept teensy 0-3.3V I/O output. From the LeCroy 688AL datasheet:

