# Lab2: Sine Wave Generation via Modulated PWM to drive a MOSFET Gate Controller for AC Inverters and Motor Controllers

## Introduction and Pre-lab

### Safety First and Operational Guidelines

In this lab, you will work exclusively with simulated signals and low-voltage circuits. **Note that the sine wave generator you design will not be connected to any MOSFETs, power electronics, or high-power loads.** However, if you experiment with MOSFETs or power electronics in other settings, it is critical to thoroughly verify all electrical connections and safety measures before connecting to motors or other high-power devices. Improper connections or inadequate precautions can lead to equipment damage, circuit malfunctions, or hazardous conditions.

When working with AC or DC circuits, **always adhere to the voltage limit of 12V for both inputs and outputs.** Exceeding this limit can result in component failure, circuit damage, or unsafe operating conditions. For example, while 24V AC is considered low voltage in many applications, it can still cause discomfort, minor shocks, or burns if improperly handled or if there is direct contact with exposed skin. Always ensure that all components are properly rated for the application, use appropriate insulation and grounding, and disconnect the power supply before making any adjustments. Your safety, and the safety of others, must always be the highest priority during any electrical work:

- **DO NOT exceed 12V** for any AC or DC circuits.
- Always verify electrical connections and component ratings before powering any circuit.
- Disconnect the power supply before making adjustments or modifications.
- Use proper insulation and grounding to prevent electrical hazards.
- If working with higher voltages or power electronics in other settings, consult with an instructor or experienced professional before proceeding.

Metal-Oxide-Semiconductor Field-Effect Transistors (MOSFETs) are widely used in power electronics due to their ability to switch high currents and voltages with high efficiency and speed. MOSFETs are voltage-controlled devices, meaning their operation is governed by the gate-to-source voltage ($V_{gs}$). When $V_{gs}$ exceeds the threshold voltage ($V_{th}$), a conductive channel forms between the drain and source, allowing current to flow. In power applications, MOSFETs are used in both low-side and high-side configurations, enabling efficient switching in circuits like DC-DC converters, inverters, and motor drives. Their low on-resistance ($R_{DS(on)}$) ensures minimal power dissipation in the conduction state, making them ideal for high-frequency switching applications where energy efficiency is critical.

In power electronics, the choice between N-channel and P-channel MOSFETs often depends on the circuit topology. N-channel MOSFETs are preferred in most applications because they have lower on-resistance and better efficiency compared to P-channel MOSFETs. However, P-channel MOSFETs are used in high-side switches for simpler gate drive circuitry in low-power systems. Driving MOSFETs effectively requires gate drivers, which provide the necessary voltage and current to switch the MOSFETs quickly, reducing switching losses and improving performance:

- An *N-channel MOSFET* is controlled by applying a positive voltage to the gate terminal relative to the source. For the MOSFET to conduct, the gate-to-source voltage ($V_{gs}$) must exceed the threshold

voltage ($V_{th}$). When $V_{gs} > V_{th}$, a conductive channel forms between the drain and source, allowing current to flow from the drain to the source. The source of an N-channel MOSFET is typically connected to a lower voltage (often ground in low-side configurations), while the drain is connected to the load or a higher voltage. The output voltage at the drain depends on the operating mode of the MOSFET: in saturation mode, it acts as a controlled current source, while in linear mode, it behaves like a resistor. Importantly, the gate must always be driven higher than the source voltage for proper switching.

- A *P-channel MOSFET* operates oppositely to an N-channel MOSFET. It conducts when the gate-to-source voltage ($V_{gs}$) is sufficiently negative ($V_{gs} < V_{th}$). For the MOSFET to switch on, the gate voltage must be lower than the source voltage. Typically, the source of a P-channel MOSFET is connected to a higher voltage (e.g., the supply voltage), while the drain is connected to the load or a lower voltage. When the gate is pulled close to the source voltage, the MOSFET turns off. The drain output voltage depends on the mode of operation, similar to the N-channel MOSFET, but the voltage polarities are reversed. P-channel MOSFETs are often used in high-side switching applications, where the source is tied to the positive supply rail.

The H-Bridge is a critical circuit topology widely used in direct current (DC) motor control, inverters, and power electronics applications. It consists of four switches, typically MOSFETs, arranged in an "H" configuration to enable bidirectional control of current through a load, such as a motor or resistive element. The design allows the polarity of the voltage across the load to be reversed, thereby enabling control over the direction of current flow and, in the case of motors, rotational direction.

Each of the four MOSFETs in the H-Bridge is assigned a specific role in current flow control. Two of these MOSFETs are positioned on the high side (e.g., switch is connected to the positive supply rail), while the other two are on the low side (connected to ground). When properly controlled, the high-side and low-side switches operate in complementary pairs—either the top left and bottom right or the top right and bottom left MOSFETs are activated—ensuring that current flows through the load in the desired direction. Simultaneous activation of both switches on the same side (high-high or low-low) is strictly avoided, as this would cause a short circuit across the supply rail, potentially damaging the circuit.

MOSFETs are preferred in H-Bridge designs due to their high switching speeds, low on-resistance, and ability to handle high currents efficiently. In high-side configurations, N-channel MOSFETs are commonly used, as they offer lower on-resistance compared to P-channel MOSFETs for the same die size. However, driving high-side N-channel MOSFETs requires specialized gate driver circuitry or bootstrap capacitors to provide a voltage higher than the source potential. The low-side MOSFETs are simpler to drive, as their sources are directly connected to ground, allowing straightforward gate control.

Flyback diodes, also known as freewheeling diodes, are essential components in an H-Bridge circuit to manage the inductive kickback generated when the current through an inductive load (e.g., a motor or solenoid) is abruptly interrupted. These diodes are placed in parallel with each MOSFET, oriented such that they do not conduct during normal operation. When a switch is turned off, the inductive load attempts to maintain the current flow, generating a high reverse voltage that can damage the MOSFETs. The flyback diodes provide a path for the current, safely dissipating the energy and protecting the circuit.

The timing and coordination of the MOSFET switching are critical to achieving efficient and reliable operation of the H-Bridge. A common strategy is to incorporate a dead-time interval between the switching of complementary MOSFET pairs to prevent shoot-through conditions. During this dead time, both MOSFETs in a pair are momentarily turned off, allowing the circuit to stabilize. This delay, although brief, must be carefully calculated to minimize switching losses and avoid compromising the overall performance of the H-Bridge.

A DC-AC inverter is an electronic device that converts DC into alternating current (AC). It is commonly used in applications such as solar energy systems, uninterruptible power supplies (UPS), and motor drives. The inverter typically employs a pulse-width modulation (PWM) technique to synthesize an AC waveform from a DC power source. The process can be broken down as follows:

The DC-AC inverter works by converting DC into AC using high-speed switching, PWM, and filtering. PWM is a technique where the width of pulses is modulated to control the average power delivered to a load. This is accomplished by turning the power switches (e.g., MOSFETs or IGBTs) on and off at a high frequency. Gate drivers are essential for driving the MOSFETs in the inverter because they ensure proper

voltage levels, fast switching, isolation, and circuit protection. Without gate drivers, the inverter would not function efficiently or reliably.

PWM operates with two key parameters: *period/frequency* and *duty cycle*:

1. **Period/Frequency**: The period of the PWM signal is the time it takes for one complete cycle of the waveform, consisting of an ON and an OFF state. The frequency is the reciprocal of the period, defined as $f = \frac{1}{T}$, where $T$ is the period. A higher frequency results in smoother output after filtering, which is critical for generating an AC-like waveform.

2. **Duty Cycle**: The duty cycle refers to the percentage of one PWM period during which the signal is ON. It is calculated as:
$$\text{Duty Cycle} = \frac{\text{ON Time}}{\text{Total Period}} \times 100\%.$$
By varying the duty cycle, the effective voltage or power delivered to the load can be controlled. For example, a 50% duty cycle delivers half the maximum possible power.

**Alignment Techniques in PWM**

PWM signals can be generated in different alignment modes to optimize the inverter's performance:

- **Left-Aligned PWM**: In left-aligned PWM, the ON time of the signal starts at the beginning of each period, and the pulse width increases or decreases to vary the duty cycle. This alignment results in a signal where all pulses begin at a fixed reference point.

- **Center-Aligned PWM**: In center-aligned PWM (or symmetric PWM), the ON time is symmetrically distributed around the center of the period. This reduces harmonics in the output waveform, making it particularly suitable for applications like motor control or grid-tied inverters, where smooth waveforms are critical.

The PWM signal, when applied to the inverter's switches, creates a high-frequency square waveform that approximates the desired AC waveform. After filtering (using LC or RC filters), the high-frequency components are removed, leaving a smooth sinusoidal output.

By carefully modulating the duty cycle of the PWM signal, the inverter can control the amplitude and frequency of the AC output to meet specific application requirements, such as matching grid standards or driving motor loads efficiently. This synthesis allows us to create AC waveform from a DC power source:

1. **DC to High-Frequency Switching:**
   - The inverter's core circuit typically involves **power switches** (e.g., MOSFETs or IGBTs) arranged in an **H-bridge configuration**.
   - The H-bridge alternates the polarity of the DC supply across the load, creating a square wave.
2. **PWM for Sine Wave Approximation:**
   - The square wave is modified using **pulse-width modulation (PWM)**.
   - In PWM, the switches are turned on and off at high frequencies, and the duty cycle of the pulses is varied to approximate a sine wave.
3. **Filtering:**
   - The output of the inverter is passed through a **low-pass filter** (inductors and capacitors) to smooth the high-frequency PWM waveform into a pure sine wave.
4. **Control:**
   - MCUs, FPGAs, or DSPs can control the switching pattern to regulate the amplitude and frequency of the AC output.

**Gate drivers** are circuits that control the operation of the MOSFETs in the inverter. They are essential for several reasons:

1. **Voltage Level Requirements:**
    - To turn a MOSFET fully on, the **gate-to-source voltage ($V_{gs}$)** must exceed a certain threshold (e.g., 10–15V for many MOSFETs).
    - The control signals from microcontrollers or DSPs typically operate at low voltages (e.g., 3.3V or 5V), which are insufficient to drive the MOSFETs directly. Gate drivers act as voltage level shifters to provide the necessary $V_{gs}$.
2. **Fast Switching:**
    - High-speed switching is critical for efficient PWM operation. The MOSFET gate has **capacitance**, and the gate driver provides the required high current to charge and discharge this capacitance quickly.
3. **Isolation:**
    - In an H-bridge configuration, the high-side MOSFETs' source terminals "float" because they are connected to the AC output. A **gate driver with isolation** ensures proper operation without causing short circuits or damage.
4. **Prevention of Shoot-Through:**
    - Gate drivers often include features like **dead-time insertion** to ensure that the high-side and low-side MOSFETs in the same leg of the H-bridge are not turned on simultaneously, which would cause a short circuit.
5. **Protection Features:**
    - Gate drivers often include built-in protections like overcurrent, overvoltage, and under-voltage lockout to safeguard the inverter circuit.

In this lab, we aim to leverage SystemVerilog on the DE10-Lite FPGA board to generate a modulated PWM (Pulse Width Modulation) signal that will interface with a Gate Controller, such as the `2110 series`, which is responsible for driving MOSFETs and powering an AC Inverter. We aim to achieve high-frequency, low-latency PWM output with fine-grained control over the duty cycle and modulation. This will ensure optimal switching of the MOSFETs in the Gate Controller, enabling efficient operation of the AC Inverter and ensuring reliable conversion of DC power to a stable AC output for various load applications.

Sine wave generation plays a vital role in a wide range of applications, particularly in power electronics and energy systems. Inverters, which convert DC power to AC, rely on pure sine waves for efficient and reliable operation. These are essential in renewable energy systems like solar and wind power, as well as in uninterruptible power supplies (UPS) for critical systems, such as medical equipment and servers. Industrial automation also benefits from sine wave generation in variable frequency drives (VFDs), which control motor speed and torque in machinery, conveyors, and robotics, while ensuring energy efficiency and precise control.

Consumer electronics and telecommunication systems also depend on sine wave inverters for stable and distortion-free AC power. Portable inverter generators, for instance, provide clean power for sensitive electronics like laptops and medical devices. Telecommunication systems use sine waves for signal transmission and powering remote stations, ensuring uninterrupted communication. In addition, testing and calibration equipment, such as oscilloscopes and waveform generators, rely on precise sine wave outputs for validating and optimizing electronic devices and circuits.

Sine wave generation is also crucial in audio systems, medical devices, and home applications. Signal generators and analog synthesizers produce sine waves for testing audio equipment and creating sound in music production. Medical devices, including imaging systems and life-support machines, require stable and noise-free AC power for accurate and uninterrupted operation. Furthermore, backup power systems and smart appliances use sine wave inverters to ensure reliable and efficient operation during power outages, protecting sensitive equipment and maintaining seamless functionality.

The PWM signal generation process in our FPGA involves creating a digital design that translates duty cycle modulation inputs into appropriate pulse sequences. This modulation can include sinusoidal,

trapezoidal, or space vector modulation techniques, depending on the AC inverter's application requirements. For example, sinusoidal PWM (SPWM) involves comparing a high-frequency triangular carrier wave with a sinusoidal reference wave to generate the duty cycle, thereby ensuring smooth and low-distortion AC waveform generation. By parameterizing the design in SystemVerilog, the PWM characteristics such as frequency, phase offset, and amplitude can be dynamically controlled, enabling adaptability to various load conditions and operational requirements.

To feed the modulated PWM signal to the Gate Controller driving the MOSFETs, the signal's integrity and timing must be maintained. This requires careful synchronization of the FPGA-generated PWM signals with the Gate Controller's input requirements. Using SystemVerilog, we can implement sophisticated clock management techniques, including clock domain crossing synchronization, to ensure the signal remains stable under varying load and environmental conditions. Additionally, the design can incorporate safety features such as dead-time insertion between switching signals to prevent shoot-through currents in the inverter's MOSFET bridge. These features are critical for ensuring both the reliability of the system and the longevity of the power electronics.

The 2110 chip family and variants like **IR2113S chip** pure sine inverter generator chip is a specialized controller designed for high-precision pure sine wave inverter applications. It integrates advanced features such as dead zone control, SPWM sine generation, amplitude modulation, soft start functionality, and real-time monitoring, making it highly suitable for efficient and reliable inverter designs. These features ensure high efficiency, low distortion, and robust operation in a variety of inverter applications such as AC inverters and AC motor controllers.

**Dead Zone Control** A **dead zone** (or dead time) is a brief delay introduced between the turn-off of one MOSFET and the turn-on of the complementary MOSFET in a half-bridge or full-bridge inverter circuit. This is crucial to prevent *shoot-through currents*, which occur when both MOSFETs in a bridge leg are conducting simultaneously, leading to short-circuit conditions across the power supply. The IR2113S provides programmable dead time settings, offering precise control over this delay with options of **300ns, 500ns, 1.0µs, and 1.5µs**. These selectable dead times allow the designer to optimize for the switching characteristics of different MOSFETs, balancing efficiency and protection. For example, higher dead times may reduce the risk of shoot-through but increase switching losses and harmonic distortion, while lower dead times improve efficiency but require faster and more precise MOSFET switching.

**High Precision, Distortion, and Harmonics Mitigation** The **IR2113S chip** is engineered to generate highly accurate sine wave outputs by leveraging **high-precision SPWM (Sinusoidal Pulse Width Modulation)** techniques. By minimizing distortion in the output waveform, the chip addresses critical issues such as harmonic distortion, which can lead to inefficiency, noise, and overheating in AC loads like motors or transformers. The integrated SPWM sine generator ensures a smooth sinusoidal waveform, which is essential for sensitive devices and applications requiring minimal Total Harmonic Distortion (THD). Furthermore, this precision results in better power quality, improved energy efficiency, and longer lifespan for connected equipment.

**SPWM Sine Generator, Dead Time Control, Amplitude Factor Multiplier, and Soft Start Circuit**

- **SPWM Sine Generator**: The SPWM generator in the 2113S modulates the width of high-frequency pulses to approximate a pure sine wave. By comparing a sinusoidal reference signal with a triangular carrier wave, the chip generates a PWM signal whose duty cycle varies sinusoidally. This enables efficient inverter operation while maintaining a low distortion sine wave output.
- **Dead Time Control Circuit**: Integrated dead time control ensures precise timing between switching signals for the high-side and low-side MOSFETs. By eliminating shoot-through currents and controlling switching transitions, the circuit increases reliability and reduces power losses.
- **Amplitude Factor Multiplier**: This feature dynamically adjusts the output sine wave's amplitude based on input or user-specified parameters. For example, it can maintain a consistent output voltage

despite fluctuations in the DC input supply or load variations, ensuring stable operation for connected devices.

- **Soft Start Circuit**: The soft start feature gradually ramps up the output voltage when the inverter is powered on, avoiding large inrush currents that could damage components or trip protection circuits. This is particularly useful for inductive loads like motors or transformers, which draw high startup currents.

**Dead Time Settings** The **IR2113S chip** offers four preset dead time options:

- **300ns**: Low dead time for faster switching MOSFETs, higher efficiency but requires precise tuning.
- **500ns**: Moderate dead time for typical applications balancing efficiency and protection.
- **1.0µs**: Suitable for MOSFETs with slower switching speeds or higher power applications.
- **1.5µs**: Maximum dead time for high-power or less precise switching scenarios.

These settings can be selected based on the MOSFET type, switching frequency, and system requirements to ensure an optimal trade-off between efficiency and protection.

**Serial Communication for Parameter Configuration** The **IR2113S chip** includes a serial communication interface, enabling external controllers or systems to configure key parameters such as:

- **Output Voltage**: Allows adjustments to match load requirements or compensate for input variations.
- **Output Frequency**: Configurable for applications needing different AC frequencies, such as 50Hz or 60Hz.
- **Additional Parameters**: Other inverter settings, like SPWM modulation depth or safety thresholds, can also be configured via the serial interface.

This programmability enhances flexibility and enables integration into a wide range of applications with varying operational demands.

**External Serial Port and LCD Display** An **external serial port** on the IR2113S chip supports communication with a **128x32 liquid crystal display (LCD) module**, providing real-time monitoring and feedback. The display can show:

- **Voltage**: Real-time AC output voltage of the inverter.
- **Frequency**: Current operating frequency, such as 50Hz or 60Hz.
- **Temperature**: Internal or external temperature readings to monitor thermal conditions.
- **Current**: Output current drawn by the load.

This monitoring capability allows users to track the inverter's performance and detect potential issues like overloading or overheating. It also facilitates maintenance and debugging, making the system more user-friendly and reliable.

# Python Code for Sine Wave Simulation from Modulated PWM

Please study and enhance below Python code that simulates a sine-wave modulated PWM to help you create similar SystemVerilog-based implementations. The code generates and visualizes signals, such as fixed-duty-cycle PWM as a reference signal, sine-modulated PWM, and a filtered output.

```python
import numpy as np
import matplotlib.pyplot as plt
from scipy.signal import lfilter

# Our simulation parameters
CLK_FREQ = 50_000_000  # 50 MHz
PWM_FREQ = 1_000       # 1 kHz
COUNTER_MAX = 49999    # For our 1kHz PWM based on 50 MHz master clock
SINE_SAMPLES = 20      # 20 points per sine wave (the more the better)
SIM_TIME = 0.100       # 100ms simulation (sine wave cycle)
STEP = 1e-7            # 0.1µs resolution
PLOT_TIME = 0.100      # Time to plot for PWM signals (based on priods)
RC_FILTER = 0.005      # RC time constant for low-pass filter

# Create time array
t = np.arange(0, SIM_TIME, STEP)

# Sine wave LUT (that will matche our SystemVerilog values)
sine_lut = np.array([
    24999, 32999, 39999, 44999, 47499, 47499, 44999, 39999,
    32999, 24999, 16999, 9999, 4999, 2499, 2499, 4999,
    9999, 16999, 24999, 32999
], dtype=np.float64)

# Calculate period and phase
period = 1/PWM_FREQ
phase = np.floor(t / period).astype(int) % SINE_SAMPLES

# Generate duty cycles
d0_duty = 24999 / (COUNTER_MAX + 1)  # Fixed 50% duty cycle
d1_duty = sine_lut[phase] / (COUNTER_MAX + 1)

# Generate PWM signals
d0 = (t % period) < (d0_duty * period)
d1 = (t % period) < (d1_duty * period)

# Apply low-pass filter to D1 to create D2 (in reallife, we can ue RC filter etc.)
alpha = STEP / (RC_FILTER + STEP)  # Filter coefficient
b = [alpha]
a = [1, -(1 - alpha)]
d2 = lfilter(b, a, d1.astype(float))

# Convert to voltage levels
d0 = d0.astype(np.float32)
d1 = d1.astype(np.float32)

# Create figure with subplots
plt.figure(figsize=(12, 10))

# Plot D0 PWM (first few periods)
ax1 = plt.subplot(4, 1, 1)
ax1.plot(t[t < PLOT_TIME], d0[t < PLOT_TIME], 'b')
ax1.set_title(f'D0 - Fixed 50% Duty Cycle PWM (First {PLOT_TIME*1000:.0f}ms)')
ax1.set_ylabel('Voltage')
ax1.set_yticks([0, 1])
ax1.grid(True)

# Plot D1 PWM (first few periods)
ax2 = plt.subplot(4, 1, 2)
```

```
ax2.plot(t[t < PLOT_TIME], d1[t < PLOT_TIME], 'r')
ax2.set_title(f'D1 - Modulated PWM (First {PLOT_TIME*1000:.0f}ms)')
ax2.set_ylabel('Voltage')
ax2.set_yticks([0, 1])
ax2.grid(True)

# Plot D1 signal envelope
ax3 = plt.subplot(4, 1, 3)
ax3.plot(t, d1_duty, 'g')
ax3.set_title('D1 - PWM Duty Cycle Envelope (Sine Simulation)')
ax3.set_xlabel('Time (s)')
ax3.set_ylabel('Duty Cycle')
ax3.grid(True)

# Plot D2 filtered output
ax4 = plt.subplot(4, 1, 4)
ax4.plot(t, d2, 'm')
ax4.set_title(f'D2 - Filtered Output (RC = {RC_FILTER*1000:.1f}ms)')
ax4.set_ylabel('Voltage')
ax4.grid(True)

plt.tight_layout()
plt.show()
```

Here is a sample output of the code:
Let's discuss each major code sections:

**1. PWM Configuration:**

- The parameters `CLK_FREQ`, `PWM_FREQ`, and `COUNTER_MAX` define the clock rate and PWM resolution:
  - `CLK_FREQ = 50 MHz`: The system clock frequency.
  - `PWM_FREQ = 1 kHz`: The desired PWM frequency.
  - `COUNTER_MAX = 49999`: The counter value for a 1 kHz PWM. It's derived as:

$$\text{COUNTER\_MAX} = \frac{\text{CLK\_FREQ}}{\text{PWM\_FREQ}} - 1$$

**2. Sine Wave Lookup Table (LUT):**

- The LUT is a 20-point sine wave represented as duty cycle values scaled to fit the PWM counter range (0 to COUNTER_MAX):

$$\text{sine\_lut} = \text{COUNTER\_MAX} \times \frac{\sin(2\pi n/20) + 1}{2}$$

- Here, $n$ ranges from 0 to 19 (20 points per cycle). The formula shifts and scales the sine wave to match the range $[0, \text{COUNTER\_MAX}]$, ensuring it stays within the bounds of the PWM duty cycle.
- We made the values precomputed and hardcoded into the array `sine_lut`, making it easily portable to your future SystemVerilog code. This ensures that the same sine wave data can be used for both simulation and hardware implementation.
- Example values from `sine_lut`:

$$\text{sine\_lut} = [24999, 32999, 39999, 44999, 47499, 47499, 44999, \dots]$$

These values represent duty cycles proportional to the sine wave at discrete points.

**3. Duty Cycle Modulation:**

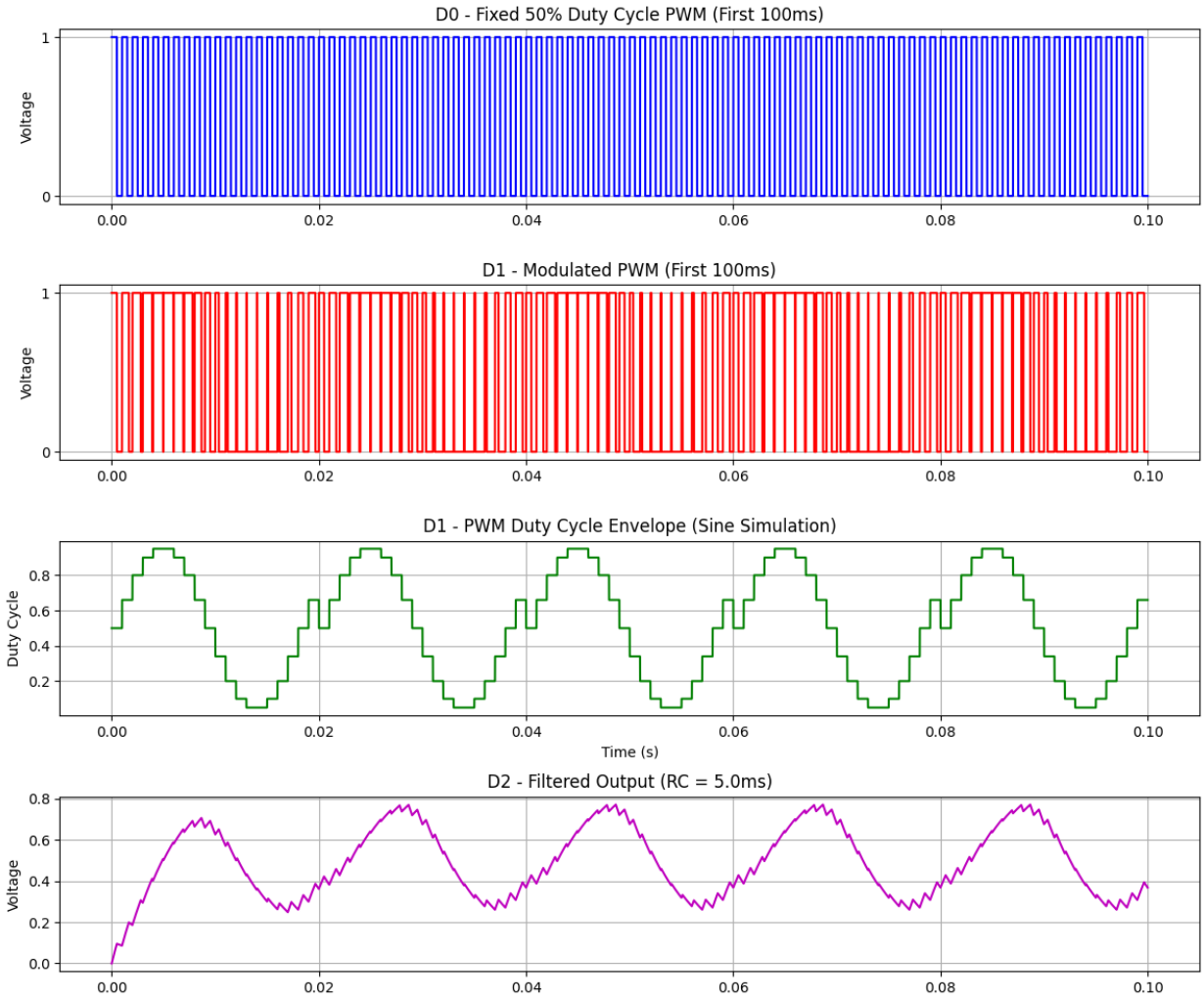


Figure 1: Sine Wave Simulation Output

- For the fixed PWM output (`D0`), the duty cycle is constant at 50%:

$$\text{d0\_duty} = \frac{24999}{\text{COUNTER\_MAX} + 1}$$

- For the sine-modulated PWM (`D1`), the duty cycle varies according to the sine LUT:

$$\text{d1\_duty} = \frac{\text{sine\_lut[phase]}}{\text{COUNTER\_MAX} + 1}$$

- The `phase` determines which index of the LUT is used at each moment, calculated as:

$$\text{phase} = \lfloor t/\text{period} \rfloor \mod \text{SINE\_SAMPLES}$$

  This cycles through the LUT entries periodically.

**4. PWM Generation:**

- The PWM signal is generated by comparing the current time within a period ($t \mod \text{period}$) to the scaled duty cycle ($duty \times period$):

- D0: Fixed PWM signal.
- D1: Sine-modulated PWM signal, where the duty cycle varies based on the sine LUT.

**5. Filtering:**

- A first-order RC low-pass filter is applied to smooth the modulated PWM signal (D1) into an analog sine wave:

$$\alpha = \frac{\text{STEP}}{\text{RC\_FILTER} + \text{STEP}}$$

  This aims to the make output (e.g., D2) smoother, simulating the effect of a physical low-pass filter. In hardware, you can use an external RC circuit of course

The plots generated by the code help us to verify:

- D0: A constant 50% duty cycle PWM.
- D1: A modulated PWM signal, where the duty cycle varies according to the sine LUT.
- D1 Duty Cycle Envelope: The sinusoidal variation of the duty cycle.
- D2: The smoothed analog sine wave after filtering.

Of course we can increase our sample points and we will have a smoother PWM modulated sine wave:

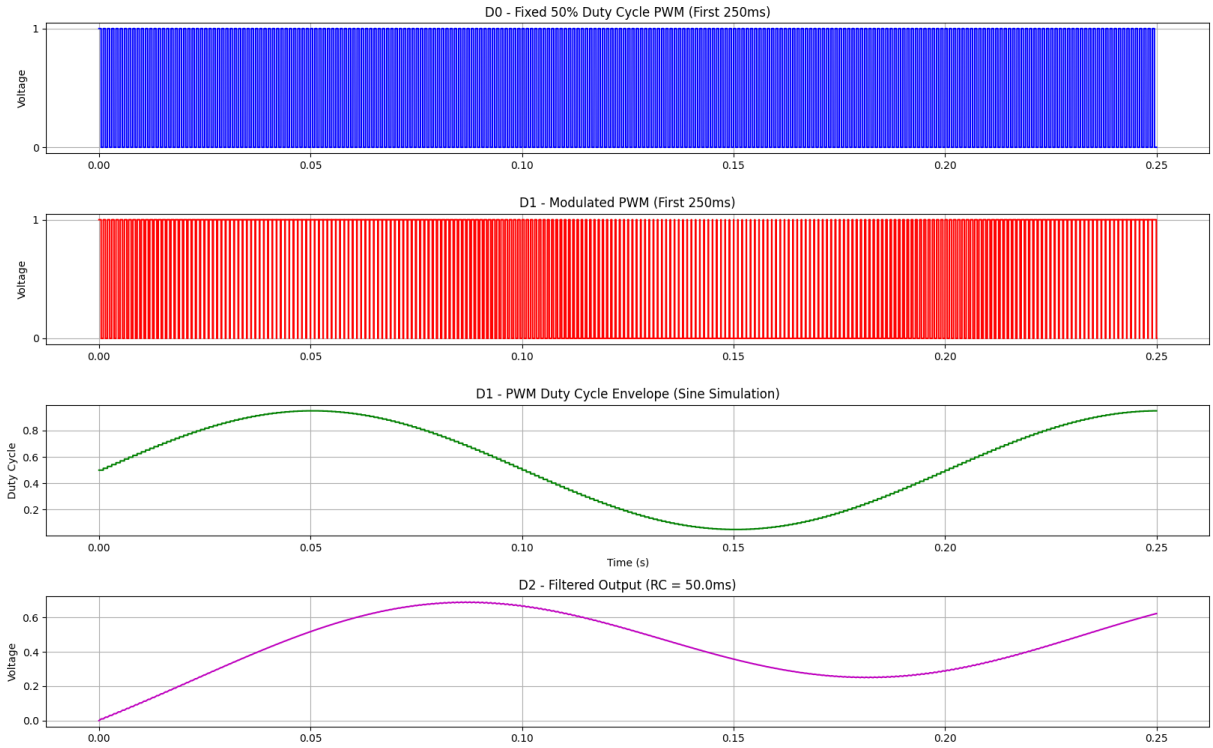


Figure 2: More Granular Sine Wave Simulation Output

After modulating the PWM to approximate a sine wave, several additional steps must be completed to implement a fully functional motor control system, especially for three-phase configurations. For instance, a three-phase motor requires synchronized sine waves with a phase difference of $\pi/3$ radians (120 degrees) between the phases. These waveforms ensure balanced operation and avoid electrical or mechanical imbalances in the motor. The synchronized PWM signals are then fed to a low-pass filter circuit. The filtering process smooths the high-frequency switching harmonics inherent in PWM signals, yielding a clean sinusoidal waveform necessary for driving the motor efficiently and minimizing harmonic distortion. This step is critical to ensure that the motor operates smoothly without vibrations, torque ripple, or excess heating caused by waveform irregularities.

Once the PWM signals are generated, they must be routed to gate driver circuits that control the switching of MOSFETs in the inverter. Gate drivers serve a crucial role in ensuring that the MOSFETs switch correctly and efficiently. They provide the necessary current to drive the MOSFET gates, control switching speeds to balance losses and EMI, and prevent shoot-through conditions where both high- and low-side MOSFETs conduct simultaneously. Proper gate driving is essential to maintain the integrity of the inverter output and to avoid failures or inefficiencies. Furthermore, the MOSFETs in the inverter must be protected by appropriate snubber circuits, flyback diodes, and thermal management systems to handle the stresses associated with high-speed switching and high current loads.

For advanced AC motor control, the PWM generation process becomes significantly more sophisticated. Advanced techniques like Space Vector PWM (SVPWM) are used to maximize the utilization of the DC bus voltage while minimizing harmonic distortion. SVPWM achieves superior waveform quality compared to traditional sinusoidal PWM methods, making it ideal for three-phase motor applications. Additionally, the system may incorporate real-time feedback from the motor, such as rotor position, phase currents, and voltages. This feedback is vital for implementing Field-Oriented Control (FOC), which enhances motor performance by aligning the stator magnetic field with the rotor position. FOC allows for precise control of torque and speed, reduces acoustic noise, and improves the overall efficiency of the motor under varying load conditions.

The integration of feedback mechanisms in motor control systems introduces new challenges and considerations. Sensors such as encoders for position feedback, current sensors for phase current monitoring, and voltage sensors for DC-link and phase voltage measurement must be chosen carefully to ensure accuracy and reliability. These sensors are critical for advanced control algorithms like FOC and for implementing safety features such as overcurrent, overvoltage, and thermal protection. Advanced motor control systems often include temperature monitoring to prevent overheating of the motor or inverter components, especially under high-load conditions. These enhancements enable robust operation even in demanding industrial or automotive applications.

In addition to the control and safety features, the quality of the inverter output depends on efficient harmonic mitigation strategies. Low-pass LC filters are essential for removing high-frequency switching harmonics, ensuring clean sinusoidal output voltage. For three-phase systems, balanced and symmetrical filter design is crucial to maintain phase synchronization and minimize distortions that could lead to unbalanced currents in the motor. The choice of filter components, such as inductors and capacitors, is driven by the switching frequency and the power rating of the system. High-speed MOSFETs, such as Silicon Carbide (SiC) or Gallium Nitride (GaN) devices, are particularly advantageous in this context as they allow for higher PWM switching frequencies, which shift harmonics to ranges that are easier to filter and reduce filter component sizes.

However, for this lab, the scope is intentionally limited to the foundational aspects of PWM generation and simple inverter control. While these topics provide a critical foundation for understanding motor control systems, advanced concepts like SVPWM, FOC, and three-phase AC motor controls are beyond the current focus. The goal is to develop a strong understanding of the fundamental principles, such as generating synchronized PWM signals, feeding them to gate driver circuits, and ensuring proper operation of a basic inverter. These steps lay the groundwork for future exploration of advanced techniques.

If you are interested in exploring further you can study advanced three-phase inverter topologies, brushless DC motor control, and sophisticated AC motor control strategies. These topics involve integrating more complex feedback mechanisms, implementing advanced PWM algorithms, and considering thermal, electrical, and mechanical optimization techniques. A good start will be to watch the webinars Texas Instruments published for their MCUs specialized in motor control application here

## Lab Objective

This lab provides hands-on experience in designing and implementing a digital sine wave generator using **Pulse Width Modulation (PWM)** on an FPGA platform, leveraging a **Category 2 Time-based Mealy Finite State Machine (FSM)**. You will generate a sine wave using a **Look-Up Table (LUT)** created in Python (or your language of choice), implement a PWM modulator in SystemVerilog or VHDL, and integrate these components into a fully functional system. The lab emphasizes **hardware-software co-design**, demonstrating how software-generated data points (such as the sine-wave LUT) are utilized by the FPGA to enable efficient signal processing. These components form the foundational building blocks for generating raw, unfiltered sine waves, which can later be refined for more complex applications.

Through this lab, you will gain practical insights into **fixed-point arithmetic**, **state machine design**, and **PWM modulation techniques**. You will explore the trade-offs between precision and resource utilization in digital systems, while developing skills in **FPGA synthesis** and **verification**. By the end of the lab, you will have a working understanding of how to design and implement a time-based Mealy FSM, generate and utilize LUTs for waveform synthesis, and create a PWM-modulated sine wave generator. These skills are essential for advanced digital signal processing and embedded systems design.

The goal is to generate:

1. A PWM signal on Arduino pin `D0`.
2. A sine wave via modulating the PWM signal on Arduino pin `D1` to drive a MOSFET gate driver for an inverter or AC motor control applications.

# Lab Tasks

**Task 1: Generating a PWM Signal**

We know that Pulse Width Modulation (PWM) is a technique where the duty cycle of a signal can be varied to control the average voltage. For a 1 kHz PWM signal:

- The DE10-Lite's clock frequency is 50 MHz.
- To achieve a 1 kHz PWM, the counter must count to COUNTER_MAX $= \frac{\text{Clock Frequency}}{\text{PWM Frequency}} - 1 = 49999$.
- The duty cycle is determined by comparing the counter value with a threshold. If the counter is less than the duty cycle value, the output is high.

Make sure you can generate above with 50% duty cycle on D0 and verifying it via Oscilloscope.

**Task 2: Modulating the PWM Signal to Simulate a Sine Wave**

The sine wave is simulated by varying the PWM duty cycle over time. This is done using:

- A **sine lookup table (LUT)** storing precomputed sine values.
- A **phase counter** that steps through the LUT at the desired sine wave frequency.
- The PWM signal is updated every PWM period to modulate its duty cycle.

Ensure you can generate a PWM-modulated signal that accurately mimics a sine wave, as demonstrated in the provided demo here .

**Step 3: Adjust Sine Wave Frequency via DE10-Lite Push Buttons**

- You will implement frequency control for the sine wave generator using the push buttons on the DE10-Lite FPGA board. The push buttons will allow you to dynamically adjust the frequency of the sine wave within a predefined range (e.g., 1kHz to 5 kHz). This is achieved by modifying the counter limit in the PWM generator, which directly affects the period of the output waveform. You will use debouncing logic to ensure reliable button inputs, preventing false triggers due to mechanical switch bouncing. The frequency adjustment logic will be integrated into the Mealy FSM, ensuring smooth transitions between frequencies without disrupting the PWM modulation.
- Additionally, you will verify the frequency adjustment functionality by observing the output waveform on an oscilloscope or logic analyzer. The LED indicators on the DE10-Lite board can also be used as a naive way to provide visual feedback of the current frequency range.

# Marking Criteria

## Precondition for Evaluation

To have your lab report graded, you must:

- Successfully demonstrate your work during the lab session.
- Answer the Teaching Assistant's (TA) questions clearly and accurately.

**Failure to demonstrate your lab or adequately respond to the TA's questions will result in a grade of zero for the lab.** Ensure that each section of your lab is thoroughly completed and meets the expectations outlined in this document. During your lab session, you must demonstrate the working components of your design to the TA. The TA will evaluate your demonstration, record the date and time, and sign your pre-lab documentation to confirm successful completion.

**Submission Requirements:**

- Lab reports, including both physical and electronic copies, must be submitted at the start of the next lab session.
- Only one report per group is allowed. Submitting multiple reports will result in a zero for all group members.

## Pre-Lab (10%)

- Submission of FSM diagrams (both Mealy and Moore) with clearly labeled states, transitions, inputs, and outputs Diagrams must be neat, accurate, and include all required information (5%).
- Pre-calculate Sine LUTs ready to be used in your HDL Codes (5%).

## Proper FSM Design and Implementation (50%)

- Correct implementation of FSM states and transitions in SystemVerilog or VHDL (10%).
- Accurate PWM modulation for Sine Wave generation (20%).
- Proper use of push buttons for frequency changes of generated PWM modulated sine wave simulation (10%).

## Testing and Verification (25%)

- Testbench demonstrating all FSM transitions and outputs with appropriate simulation waveforms (15%).
- Evidence of successful hardware testing on the DE10-Lite board, including photos or videos showcasing the working outputs (10%).
- Signed pre-lab papers from the TA confirming proper demonstrations and adequate responses to questions. Each group member must demonstrate their understanding by answering specific questions correctly and clearly.

## Lab Report (15%)

- Comprehensive formal report including:
  - FSM diagrams and detailed explanations of design choices (5%).
  - Discussion of challenges encountered and how they were resolved (5%).
  - Results from simulation and hardware testing with visual evidence (5%).
- **Submission of both electronic and physical copies is required.**

## Bonus (Up to 100%)

- **Bonus Part A (Up to 15%):** Implement Space Vector Pulse Width Modulation (SVPWM) on the FPGA to generate three-phase AC sine waves (apply your low-pass filter). Demonstrate the creation of balanced three-phase output voltages with adjustable frequency and amplitude. Validate the output by analyzing the waveforms on an oscilloscope.
- **Bonus Part B (Up to 15%):** Upon successful completion of Part A, integrate the SVPWM implementation with a three-phase inverter. Perform a simulation or utilize a hardware setup to showcase the generation of a clean sinusoidal waveform for each phase. Demonstrate waveform integrity using an oscilloscope.

- **Bonus Part C (Up to 5%):** Building on Part B, implement dead-time insertion in the SVPWM logic using DIP switches on the DE10-Lite FPGA board. Verify and demonstrate the impact of dead-time on the output waveforms using an oscilloscope.
- **Bonus Part D (Up to 25%):** Upon successful completion of Part C, integrate your wave generator with a gate driver controller of your choice. For example, you may use commercially available drivers, custom designs, or other innovative approaches. Demonstrate the ability of your system to drive the three-phase inverter effectively, ensuring accurate signal timing and control.
- **Bonus Part E (Up to 40%):** If Part D is completed successfully, develop an end-to-end system to control a brushless DC (BLDC) of your choice. Implement Field-Oriented Control (FOC) for the motor, ensuring smooth and efficient operation. Note that pre-built Electronic Speed Controllers (ESCs) with integrated FOC functionality are **NOT** permitted. Demonstrate the system's performance and efficiency through real-time operation of the motor if you choose the sensor-less designs.

## Penalties

- Late submission will result in a zero for all group members.
- Failure to sign and date the submitted pre-lab work as instructed will result in no marks for the pre-lab.
- Submitting multiple reports for a group will result in a zero for all members.

Good luck, and have fun designing!