1. **Bộ tạo ngõ ra sóng bằng digital**

**1. Giới thiệu**

Bộ tạo sóng số (Digital Waveform Generator) là một thành phần quan trọng trong nhiều ứng dụng xử lý tín hiệu số (DSP), tổng hợp âm thanh, truyền thông, và các hệ thống đo lường. Trên FPGA, việc tạo sóng số có thể được thực hiện bằng nhiều phương pháp khác nhau, trong đó phương pháp sử dụng **Bảng Tra Cứu (Look-Up Table - LUT) kết hợp với Bộ Tích Lũy Pha (Phase Accumulator - PA)** là một trong những cách tiếp cận hiệu quả nhất vì nguyên lý hoạt động đơn giản và có thể tùy biến nhiều dạng sóng mong muốn.

Phương pháp này giúp tạo ra các dạng sóng có độ chính xác cao, tần số có thể điều chỉnh linh hoạt mà không cần thực hiện các phép toán lượng giác phức tạp trong phần cứng. Với LUT, chúng ta có thể lưu trữ sẵn các giá trị mẫu lượng tử của một chu kỳ sóng và chỉ cần sử dụng địa chỉ pha từ bộ tích lũy để trích xuất giá trị tương ứng. Điều này giúp giảm đáng kể mức sử dụng tài nguyên logic trên FPGA và tăng hiệu suất xử lý nhưng điểm hạn chế là tốn hiều logic gate tài nguyên trên FPGA.

**2. Nguyên lý hoạt động**

Bộ tạo sóng số bằng phương pháp LUT và PA bao gồm ba thành phần chính:

**2.1 Bộ Tích Lũy Pha (Phase Accumulator - PA)**

Bộ tích lũy pha là một bộ đếm có bước nhảy được xác định trước (phase\_step). Giá trị của phase\_step quyết định tần số của sóng đầu ra. Trên mỗi chu kỳ xung nhịp, bộ tích lũy pha cộng thêm phase\_step vào giá trị pha hiện tại, tạo thành một pha liên tục từ 0 đến giá trị tối đa.

Công thức xác định tần số của tín hiệu đầu ra:

Fout=

Trong đó:

* phase\_step là giá trị bước nhảy pha.
* Clk là tần số xung nhịp hệ thống, ở lab này ta dùng De10 nên clk sẽ là 50Mhz.
* Số mẫu lượng tử 1 chu kì của sóng: Có thể là 1024 hoặc nhỏ hơn để tiết kiệm dung lượng logic gate.

Ví dụ: nếu clk=50MHz, phase\_step=24, mẫu lượng tử là 1024 , thì fout=1,017KHz

A diagram of a line of different signals

AI-generated content may be incorrect.Giả sử một xung được cấu hình là 1024 mẫu thì trong một chu kì 1Mns, thời gian chạy 1 mẫu sẽ là 977ns. Vậy 1 mẫu sẽ tốn 49 chu kì clock của De10. Từ đó nếu mốt có được tần số mong muốn ta chỉ cần nạp phase step từ công thức sau

Phase\_step=

Bảng nạp Phase\_step tương ứng với tần số cho ra :

|  |  |
| --- | --- |
| F\_out | Phase\_step ( thang decimal) |
| 1KHz | 49 |
| 2KHz | 24 |
| 3KHz | 16 |
| 4KHz | 12 |
| 8KHz | 6 |
| 16KHz | 3 |

A diagram of a computer program

AI-generated content may be incorrect.Schematic của module thiết kế:

Giải thuật của phase accumulator hoạt động như sau:

Module phase accumulator gồm 2 bộ conter: khi counter thứ nhất đạt đến giá trị phase\_step đầu vào thì phase\_acc mới cộng 1 để tăng địa chỉ trỏ vào LUT để lấy mẫu. Và cứ tuần tự counter 1 reset về 0 và tiếp tục tăng đến khi đạt giá trị tối đa để tăng LUT.

Tần số của sóng phụ thuộc vào giá trị phase\_step mà ta đã cho trước.

**2.2 Bảng Tra Cứu (Look-Up Table - LUT)**

LUT chứa các mẫu số đã được tính trước của một chu kỳ sóng, chẳng hạn như sóng sin. Bộ tích lũy pha sử dụng giá trị pha của nó để làm địa chỉ truy xuất LUT, từ đó lấy ra giá trị biên độ tương ứng.

* LUT có thể được triển khai bằng bộ nhớ ROM trên FPGA.
* Số lượng mẫu trong LUT càng nhiều thì độ mịn của sóng càng cao.
* LUT có thể chứa nhiều dạng sóng khác nhau, chẳng hạn như:
  + **Sóng sin** (Sine wave)
  + **Sóng vuông** (Square wave)
  + **Sóng tam giác** (Triangle wave)
  + **Sóng răng cưa** (Sawtooth wave)
  + **Sóng nhịp tim** (ECG)

Ta lượng tử dạng sóng thành 1024 mẫu với 16 bit lượng tử. Ngõ vào của LUT là phase\_acc để trỏ vào địa chỉ lấy mẫu. Địa chỉ này tăng dần và sẽ reset về 0 khi quét hết 1 bảng LUT tương đương với việc xuất ra một chu kì sóng.

**A screen shot of a computer

AI-generated content may be incorrect.Wave form của sóng sine**

Khi thay đổi phase\_step như bảng trên ta có thể thấy tần số của sóng sine thay đổi từ 1kHz lên 16kHz. Sóng sine chỉ có phần dương vì bộ codec chỉ hiểu giá trị dương.

**Wave form của xung vuông**

Xung vuông đảo trạng thái mức logic sau nửa chu kì. Nhìn mô phỏng cho thấy khi thay đổi phase\_step tần số xung vuông cũng thay đổi theo đúng như lý thuyết đã chướng minh.

A screenshot of a computer

AI-generated content may be incorrect.

**Wave form của xung tam giác**

A screenshot of a computer

AI-generated content may be incorrect.Xung tam khi tăng tuyến tính đạt đến giá trị peak thì giảm về 0 một cách tuyến tính ( chỉ có phần dương vì bộ codec chỉ hiểu giá trị dương ). Nhìn mô phỏng cho thấy khi thay đổi phase\_step tần số xung tam giác cũng thay đổi theo đúng như lý thuyết đã chướng minh.

**Wave form của xung răng cưa**

Xung răng khi đạt đến giá trị peak thì giảm về 0 ( chỉ có phần dương vì bộ codec chỉ hiểu giá trị dương ). Nhìn mô phỏng cho thấy khi thay đổi phase\_step tần số xung răng cưa cũng thay đổi theo đúng như lý thuyết đã chướng minh.

A screenshot of a video player

AI-generated content may be incorrect.

**Wave form của điện tâm đồ ecg** A diagram of a waveform

AI-generated content may be incorrect.

Một chu kì điện tâm đồ được cấu tạo gần giống các phần như là hàm gauss, hàm sine, hàm sigmoid và hàm flat top. Ta có thể dựa vào các lượng tử của các sóng này lái để điều chỉnh tần số rộng hẹp của xung, và chia tỉ lệ để ghép thành một sóng ecg với số mẫu là 1024.

A screen shot of a computer monitor

AI-generated content may be incorrect.Kết luận lái và ghép các lượng tử cho ra sóng ecg như yêu cầu và có thể chỉnh tần số dựa vào phase step tương tự như các sóng ở trên.

**2.3. Kiểm thử và Triển khai**

* **Mô phỏng trên Xcelium:** Kiểm tra tín hiệu đầu ra chế độ analog nhận thấy sóng đạt đúng yêu cầu.
* **Tổng hợp & Nạp lên FPGA:** Kết nối FPGA với DAC để hiển thị tín hiệu trên máy hiện sóng.
* **Điều chỉnh tần số sóng:** Thay đổi giá trị phase\_step để kiểm tra khả năng tạo tần số khác nhau.

**2.4. Kết luận**

Phương pháp này cung cấp một cách tiếp cận hiệu quả để tạo sóng tín hiệu trên FPGA với độ chính xác cao và hiệu suất tốt. LUT giúp giảm tải tính toán, trong khi bộ tích lũy pha đảm bảo điều chỉnh tần số dễ dàng. Việc triển khai trên FPGA có thể mở rộng để hỗ trợ nhiều dạng sóng khác nhau, phục vụ đa dạng ứng dụng trong xử lý tín hiệu số.

1. **Tạo noise bằng LFSR**

**1. Giới thiệu**

Nhiễu là một thành phần quan trọng trong nhiều ứng dụng điện tử và viễn thông, bao gồm kiểm tra hệ thống, mã hóa, và mô phỏng tín hiệu ngẫu nhiên. Một trong những phương pháp phổ biến để tạo nhiễu số trên FPGA là sử dụng Bộ Dịch Phản Hồi Tuyến Tính (Linear Feedback Shift Register - LFSR). LFSR là một bộ tạo số giả ngẫu nhiên (PRNG) hiệu quả, có thể được triển khai với tài nguyên phần cứng tối thiểu.

**2. Nguyên lý Hoạt Động Của LFSR**

LFSR là một thanh ghi dịch có phản hồi tuyến tính từ các bit cụ thể trong thanh ghi, tạo ra một chuỗi bit có chu kỳ dài trước khi lặp lại. Phản hồi tuyến tính thường được xác định bởi một đa thức sinh (polynomial tap), giúp đảm bảo độ dài chu kỳ tối đa.

**2.1 Cấu Trúc LFSR**

Một LFSR cơ bản gồm các thành phần chính:

1. Thanh ghi dịch (Shift Register): Lưu trữ trạng thái của LFSR.
2. Hàm phản hồi tuyến tính (Feedback Polynomial): Xác định cách tạo bit tiếp theo từ các bit hiện tại.
3. XOR Gate: Kết hợp các bit theo đa thức phản hồi để tạo bit mới.
4. Xung nhịp (Clock): Điều khiển quá trình dịch và cập nhật dữ liệu.

**2.2 Đa Thức Phản Hồi Tuyến Tính**

Đa thức phản hồi xác định cách các bit được kết hợp để tạo ra chuỗi số giả ngẫu nhiên. Ví dụ, một LFSR 24-bit sử dụng đa thức: có thể được triển khai như sau:

wire feedback = lfsr\_reg[23] ^ lfsr\_reg[22] ^ lfsr\_reg[21] ^ lfsr\_reg[16];

Điều này có nghĩa là bit mới của thanh ghi được tạo bằng cách XOR các bit được chọn từ các vị trí 23, 22, 21 và 16.

**3. Tạo Nhiễu Bằng LFSR**

LFSR có thể được sử dụng để tạo nhiễu số bằng cách:

* Lấy giá trị hiện tại của thanh ghi làm dữ liệu nhiễu.
* Điều chỉnh biên độ nhiễu bằng cách nhân với một hệ số điều chỉnh.
* Thay đổi tần số của nhiễu bằng cách giảm tốc độ cập nhật của LFSR (sử dụng bộ chia xung).

**3.1 Điều Chỉnh Biên Độ Nhiễu**

Để điều chỉnh biên độ của nhiễu, ta sử dụng một hệ số nhân (gain) điều khiển bởi người dùng:

always @(posedge clk\_div) begin

noise\_out <= (lfsr\_reg \* noise\_gain) >> 4; // Điều chỉnh biên độ

end

Ở đây, noise\_gain là giá trị từ 1 đến 15, cho phép điều chỉnh mức nhiễu linh hoạt.

**3.2 Điều Chỉnh Tần Số Nhiễu**

Tần số nhiễu có thể được thay đổi bằng cách sử dụng bộ chia xung (clk\_div) để giảm tốc độ cập nhật của LFSR:

clk\_div clk\_div\_inst (

.clk(clk),

.rst(~rst\_n),

.freq\_inc(noise\_freq\_inc),

.freq\_dec(noise\_freq\_dec),

.clk\_out(clk\_out)

);

Điều này giúp giảm tốc độ cập nhật LFSR, từ đó thay đổi phổ tần số của tín hiệu nhiễu.

**4. Ứng Dụng Của LFSR Trong Tạo Nhiễu**

LFSR là một phương pháp tạo nhiễu hiệu quả với nhiều ứng dụng:

* Mô phỏng môi trường nhiễu trong hệ thống truyền thông số.
* Kiểm thử hệ thống xử lý tín hiệu và mã hóa dữ liệu.
* Tạo nhiễu trong các ứng dụng âm thanh và hình ảnh số.
* Ứng dụng trong mã hóa và bảo mật dữ liệu.

**5. Kết Luận**

Sử dụng LFSR để tạo nhiễu trên FPGA là một phương pháp đơn giản, hiệu quả và tiết kiệm tài nguyên phần cứng. Với khả năng điều chỉnh biên độ và tần số, LFSR có thể đáp ứng nhiều yêu cầu khác nhau trong các hệ thống xử lý tín hiệu số.

A screenshot of a computer

AI-generated content may be incorrect.Với ngõ vào kết nối với các nút nhấn để tăng biên độ và tần số, biên độ và tần số của nhiễu cũng đã tăng theo yêu cầu.

1. **Module amplifier khuếch đại biên độ**

**1. Giới Thiệu**

Trong hệ thống xử lý tín hiệu số (DSP), bộ khuếch đại (amplifier) đóng vai trò quan trọng trong việc điều chỉnh biên độ của tín hiệu. Trong thiết kế FPGA, khuếch đại tín hiệu số có thể được thực hiện thông qua phép nhân với một hệ số khuếch đại (gain multiplier).

Hệ thống amplifier trong code trên sử dụng Multiplexer (MUX) và Booth Multiplier để điều chỉnh biên độ tín hiệu đầu vào, giúp tạo ra tín hiệu đầu ra có cường độ mong muốn. Hệ thống này cho phép thay đổi giá trị khuếch đại thông qua nút nhấn (btn0, btn1).

**2. Nguyên Lý Hoạt Động**

**2.1 Điều Chỉnh Hệ Số Khuếch Đại**

* Người dùng có thể điều chỉnh hệ số khuếch đại bằng cách nhấn các nút btn0 và btn1.
* btn1 tăng giá trị khuếch đại, trong khi btn0 giảm giá trị khuếch đại.
* Các giá trị khuếch đại khả dụng là: {1, 2, 4, 8, 16}.
* Một bộ MUX 5:1 (mux\_5to1) được sử dụng để chọn hệ số khuếch đại phù hợp dựa trên giá trị amp\_sel.

**2.2 Khuếch Đại Tín Hiệu Số**

* Sau khi chọn hệ số khuếch đại từ MUX, tín hiệu đầu vào wave được nhân với hệ số khuếch đại (mul\_const).
* Bộ nhân Booth (boothmul) thực hiện phép nhân giữa wave và hệ số khuếch đại để tạo ra amp\_out.
* Quá trình này đảm bảo việc nhân số học được thực hiện nhanh và tối ưu trên FPGA.

**2.3 Giải thuật nhân 2 số booth**

Giải thuật Booth hoạt động dựa trên việc **mã hóa nhóm hai bit liên tiếp** của số nhân để quyết định các thao tác cộng, trừ hoặc giữ nguyên. Điều này giúp giảm số lượng phép toán cần thiết khi thực hiện nhân.

Mỗi bước của thuật toán xem xét **hai bit liên tiếp** trong số nhân (multiplicand) và số mở rộng (Q0 và Q-1), theo quy tắc sau:

|  |  |  |
| --- | --- | --- |
| **Q[i]** | **Q[i-1]** | **Thao Tác** |
| 00 | - | Giữ nguyên |
| 01 | - | Cộng số bị nhân |
| 10 | - | Trừ số bị nhân |
| 11 | - | Giữ nguyên |

* Q[i] là bit hiện tại của số nhân.
* Q[i-1] là bit trước đó (bit mở rộng, ban đầu là 0).
* Dịch phải số bị nhân sau mỗi bước để cập nhật kết quả.

**Các Bước Thực Hiện**

1. Khởi tạo thanh ghi tích lũy (A) = 0 và mở rộng số nhân (Q-1 = 0).
2. Duyệt từng bit trong số nhân và thực hiện theo bảng quy tắc trên.
3. Dịch phải thanh ghi chứa tích lũy và số nhân.
4. Lặp lại cho đến khi xử lý hết các bit của số nhân.

A diagram of a computer

AI-generated content may be incorrect.**2.4. Schematic và testbench của bộ Amplifier**

1. **Top module**

De10 có 10 switch và 4 botton, do đó ta sẽ dùng các botton theo bảng như sau

|  |  |  |
| --- | --- | --- |
| Thông số điều chỉnh | Tăng | Giảm |
| Biên độ | btn1 | btn0 |
| Tần số | btn3 | btn2 |

A diagram of a circuit

AI-generated content may be incorrect.Ta sẽ dùng 4 swich để chỉnh các chế độ xuất sóng mong muốn, riêng SW4 ta dùng để điều kiển demux để chọn là điều chỉnh thông số của sóng hay nhiễu.

* 1. A diagram of a circuit

     AI-generated content may be incorrect. **Bộ mux để chọn Phase\_step để điều kiển tần số**

Ta có thể chọn các tần số 1k 2k tới 16 nhờ nhấn nút btn3 và btn2 và với SW4 chỉnh về 0. Ngõ ra nối trực tiếp vào các LUT của các wave form để tạo ra sóng.

* 1. A diagram of a circuit

     AI-generated content may be incorrect.**Bộ xử lý sóng trước khi đưa vào bộ DAC**

SW3: Dùng để select sóng có trộn nhiễu hay không

Ta có thể điều chỉnh tần số của nhiễu và sóng bằng cách lựa chọn SW4 và btn[3:0] như đã trình bày ở trên

Bảng SW2, SW1, SW0 để select tạo sóng nào ở ngõ ra bằng bộ mux 6 sang 1

|  |  |  |
| --- | --- | --- |
|  | SW2 , SW1, SW0 | SÓNG |
| 0 | 000 | Sine |
| 1 | 001 | Vuông |
| 2 | 010 | Tam giác |
| 3 | 011 | Răng cưa |
| 4 | 100 | Ecg |
| 5 | 101 | Nhiễu |