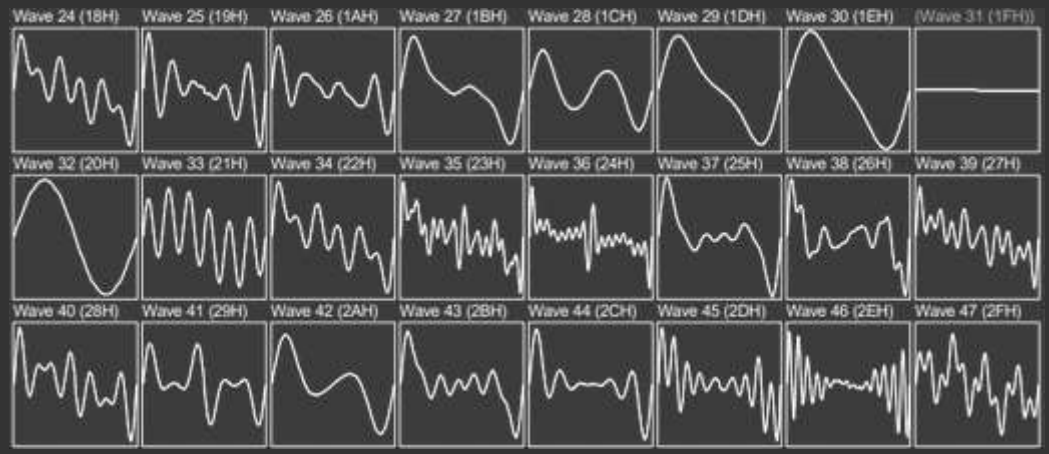
Elektroniczne instrumenty muzyczne

SYNTEZA TABLICOWA

Cyfrowe generowanie sygnałów



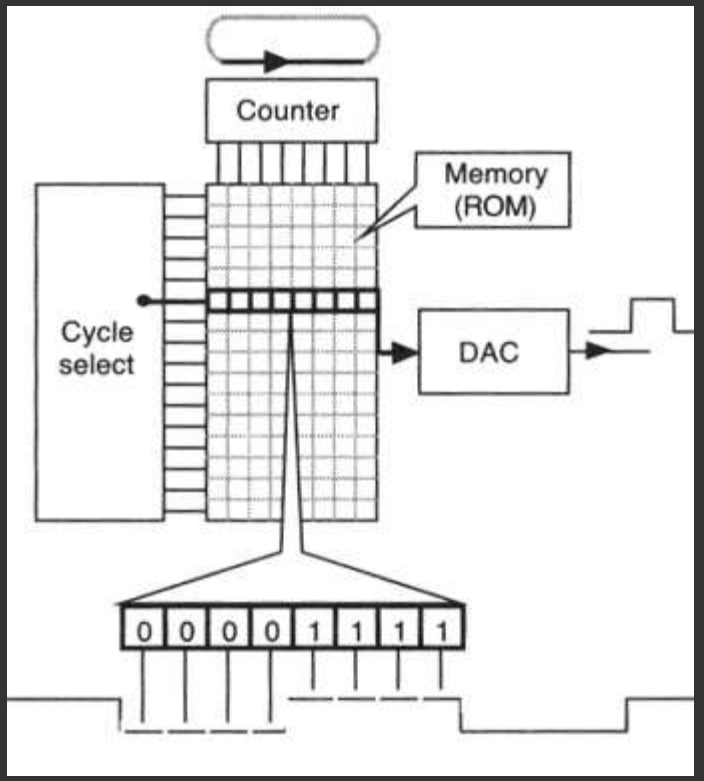
Opracowanie: Grzegorz Szwoch, Politechnika Gdańska, Katedra Systemów Multimedialnych

Analogowe i cyfrowe oscylatory

- Na początku lat 198x wciąż dominowały analogowe syntezatory subtraktywne.
- Liczba kształtów fali możliwych do wygenerowania przez analogowe oscylatory była ograniczona (piła/trójkąt/prostokąt/impuls + kombinacje).
- Cyfrowe instrumenty (właściwie wczesne samplery) istniały na rynku, ale były bardzo drogie i niedostępne dla większości muzyków.
- Idea generowania fal poprzez odczyt cyfrowych próbek z pamięci była znana, ale ówczesne pamięci ROM były drogie i miały małą pojemność (16-64 KB).

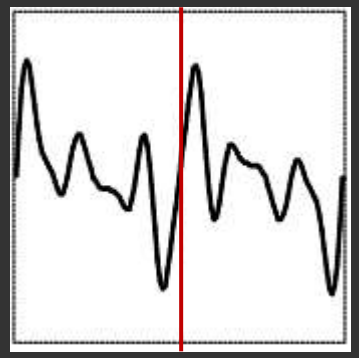
Cyfrowy oscylator - tablica fal

- Założenie: w pamięci zapisujemy próbki pojedynczego okresu fali (*single cycle oscillator*) – nie cały dźwięk (jak w samplerze).
- Odczytując te próbki „w kółko”, generujemy ciągłą falę.
- Możemy mieć w pamięci wiele różnych kształtów fali do wyboru – tablica fal (*wavetable*).
- Możliwe są praktycznie dowolne kształty fal.
- Można czytać próbki na zmianę do przodu i do tyłu, można odwracać znak próbek.
- Można nawet zmieniać kształt odczytywanej fali w trakcie generowania dźwięku - przemiatanie tablicy (*table sweep*).

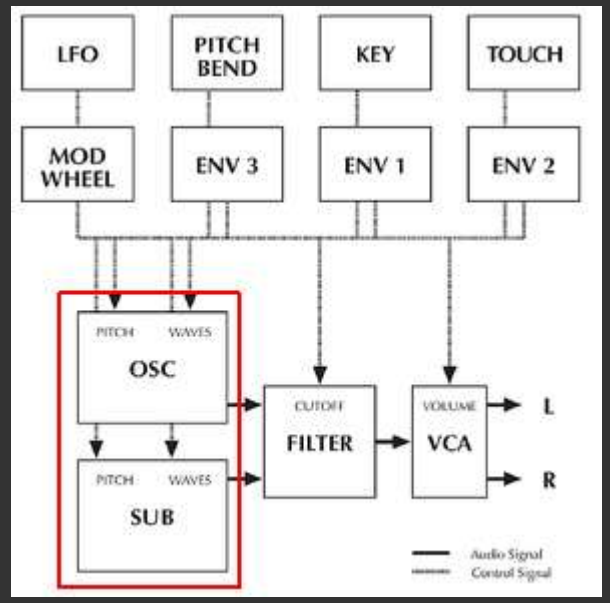


Synteza tablicowa

Synteza tablicowa (*wavetable synthesis*) została opracowana przez Wolfganga Palma (PPG) i zastosowana w syntezatorach z serii *Wave*.

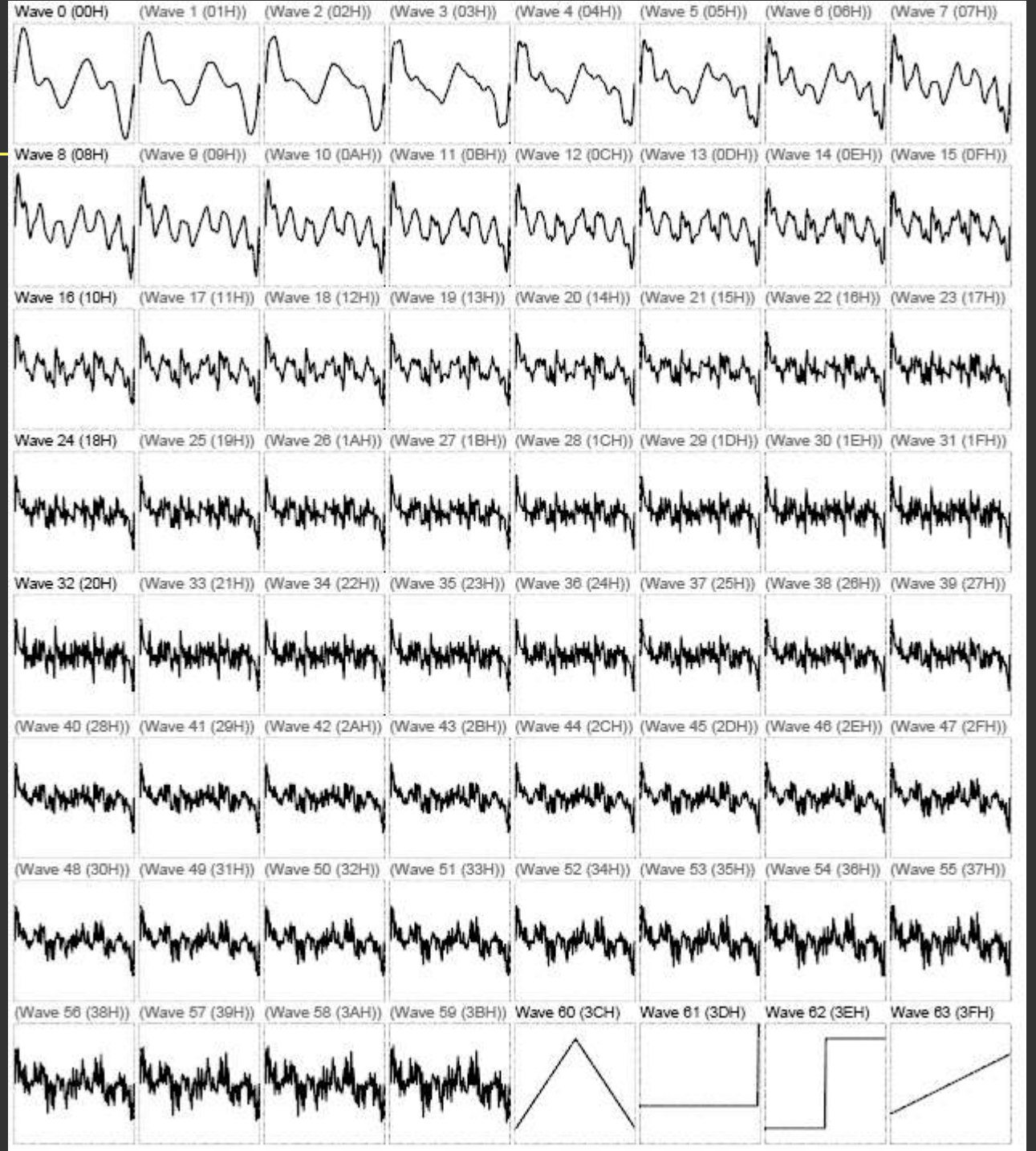


- Fale (*wave*) – okresy sygnałów zapisane w pamięci.
  256 zapisanych fal, 64 próbki na pół okresu,
  8 bitów na próbkę: 16 KB ROM na wszystkie fale.
- Tablice (*wavetable*) – grupy fal (64 w PPG).
- Odczyt fal z pamięci i ich zapętlanie.
- Modulacja indeksu fali w tablicy – zmiany brzmienia.
- Konwersja cyfrowego sygnału na analogowy.
- Dalsze przetwarzanie przez filtry VCF i modulatory LFO i EG, tak jak w syntezie subtraktywnej.



Synteza tablicowa

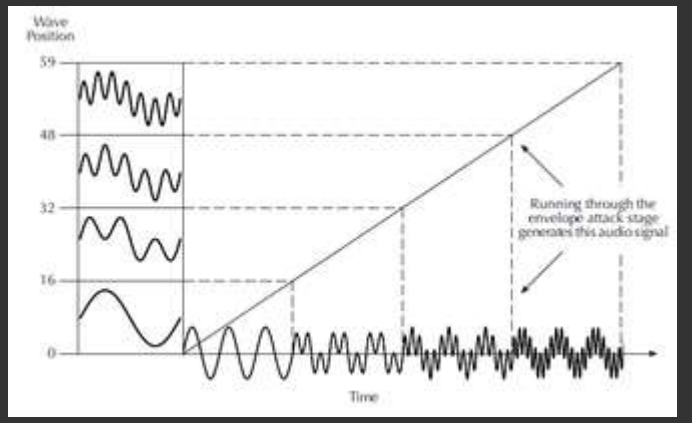
- Tablica w PPG zawiera 64 fale.
- 60 fal: zwykle ustalony kształt fali, wzrastająca liczba harmonicznych (coraz jaśniejsza barwa).
- 4 ostatnie fale: klasyczne kształty.
- 30 tablic, w tym jedna dostępna zawsze, reszta do wyboru.
- Dwa generatory, możliwość łączenia dwóch głosów (dwóch kształtów fal).

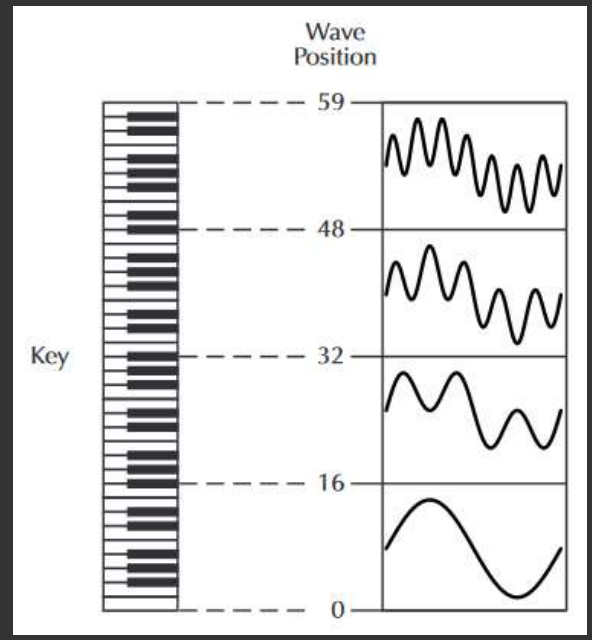


Przemiatanie tablicy

Indeks fali w tablicy może być modulowany – przemiatanie tablicy (*table sweep*), co pozwala na płynne zmiany brzmienia dźwięku:

- generator obwiedni EG
  - wartość sygnału zmienia indeks odczytywanej fali,
  - modyfikacja brzmienia w fazie ataku,
- LFO:
  - cykliczna modulacja indeksu odczytywanej fali,
  - zmiana barwy w fazie podtrzymania,
- klawiatura: bardziej złożone fale dla niższych klawiszy, mniej złożone dla wyższych – ograniczenie aliasingu,
- sterowniki, np. pokrętło *modulation*.





Instrumenty tablicowe PPG

PPG *Wave* 2 (1981), *Wave 2.2* (1982) i *Wave 2.3* (1984).

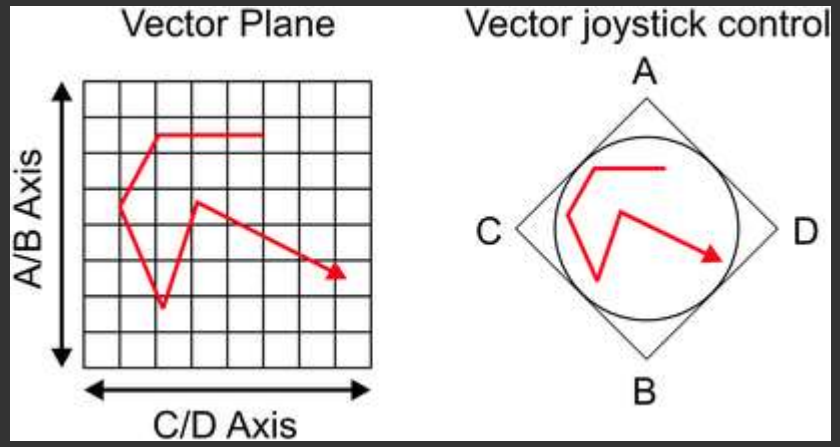
Charakterystyczne ostre, „dzwonowe” brzmienie – dużo składowych na wysokich częstotliwościach, również nieharmonicznych (efekt aliasingu i braku interpolacji).

PPG *Waveterm* – dodatkowy komputer, umożliwiał wgrywanie własnych, krótkich (do 1 sekundy) dźwięków, nazywanych transjentami (*transient*)
- uproszczony sampling.

Prophet VS – synteza wektorowa

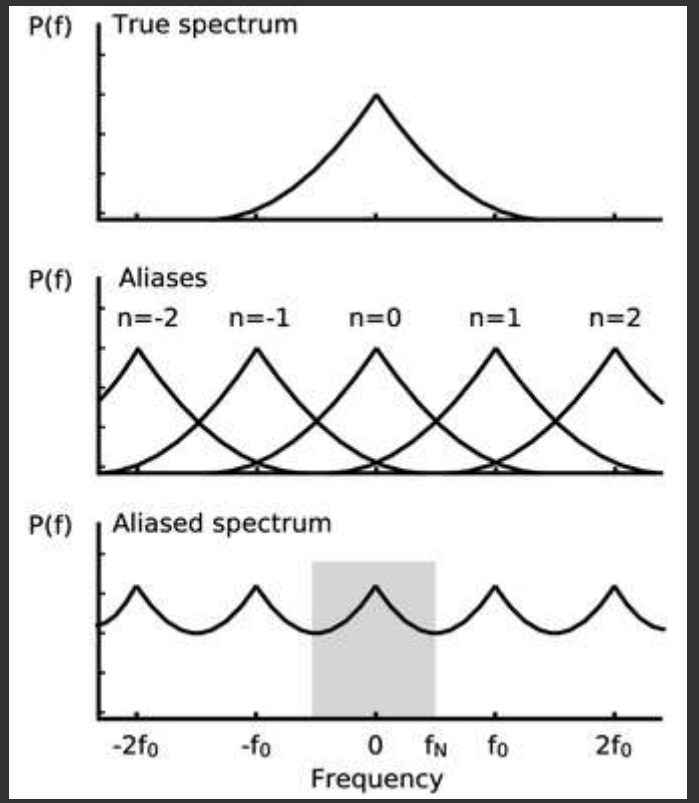
Sequential Circuits *Prophet VS* (1986)

- Zamiast tablicy – cztery kształty fal: A, B, C, D (pojedyncze cykle, 96 fal).
- Wybrane cztery fale są miksowane w proporcjach wyznaczonych przez joystick.
- Można modulować proporcje za pomocą generatorów obwiedni i LFO.
- Synteza wektorowa (*vector synthesis*) – odmiana syntezy tablicowej. Ruchy joysticka zakreślają „wektory”. W pełni cyfrowy instrument.



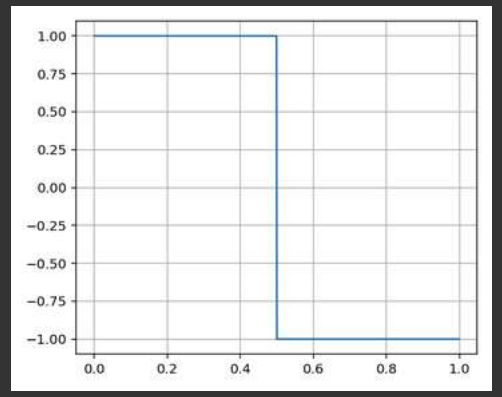
Problem aliasingu

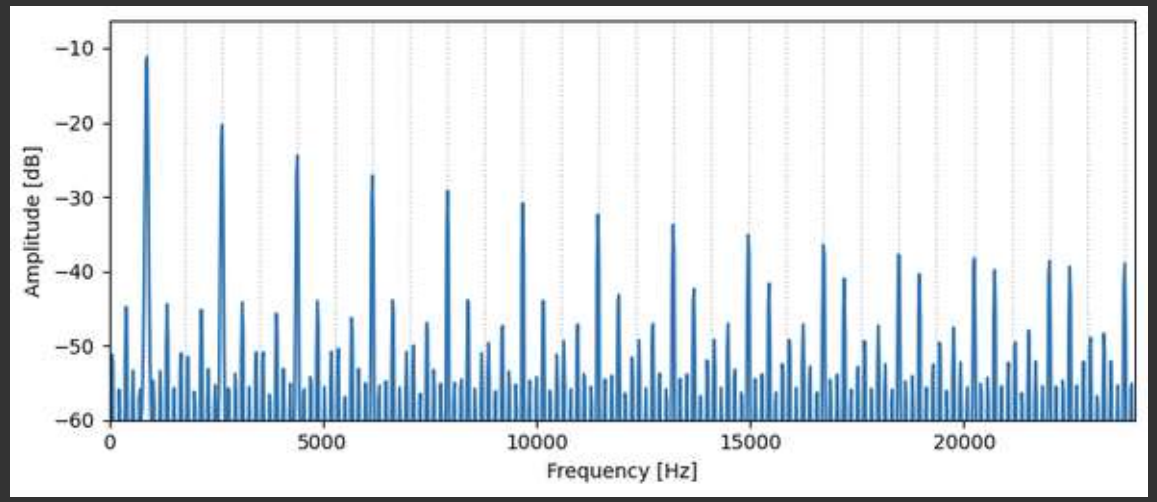
- Cyfrowe generowanie sygnałów wprowadza problem aliasingu widma, którego nie ma w technice analogowej.
- Widmo sygnału cyfrowego powtarza się okresowo co częstotliwość próbkowania *fs*.
- Sygnał rzeczywisty: dwie kopie widma (0…*fs*/2, *fs*/2…*fs*), odbicie lustrzane.
- Sygnał musi być ograniczony pasmowo (*bandlimited*) do częstotliwości Nyquista, równej *fs*/2.
- Jeżeli składowe widma wychodzą poza ten zakres, nakładają się na siebie i powstaje aliasing widma.
- Efekt „odbijania się” składowych widma od *fs*/2 i 0.



Problem aliasingu

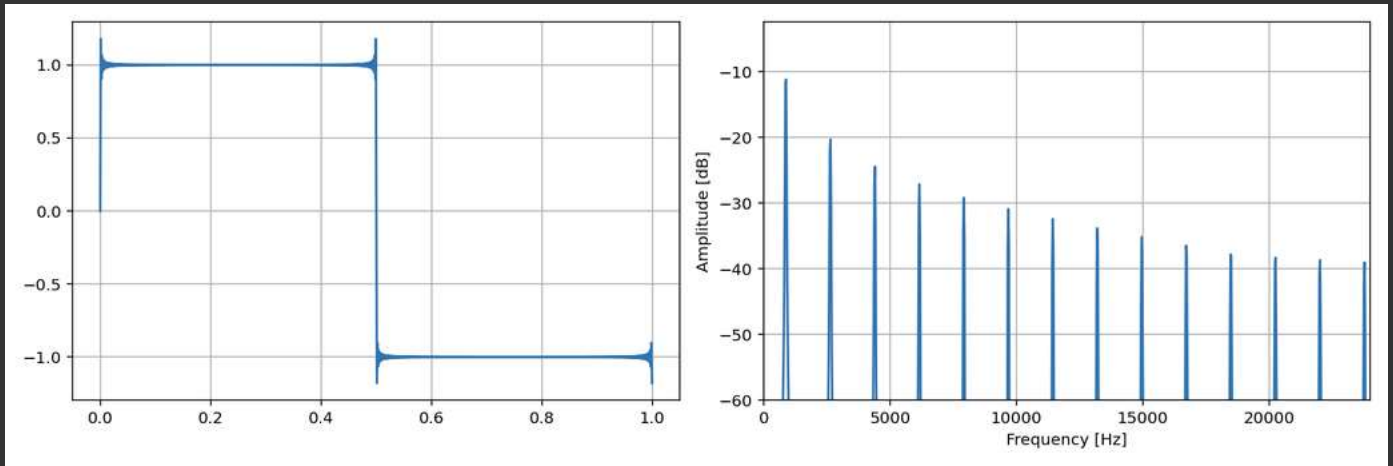
- Sygnały harmoniczne z VCO można zapisać jako szereg Fouriera, który jest nieskończony.
- Składowe widma przekraczające *fs*/2 powodują aliasing.
- „Odbite” składowe wprowadzają nieharmoniczność i psują brzmienie.
- Przykład: „naiwna” fala prostokątna – produkuje znaczny aliasing.





Problem aliasingu

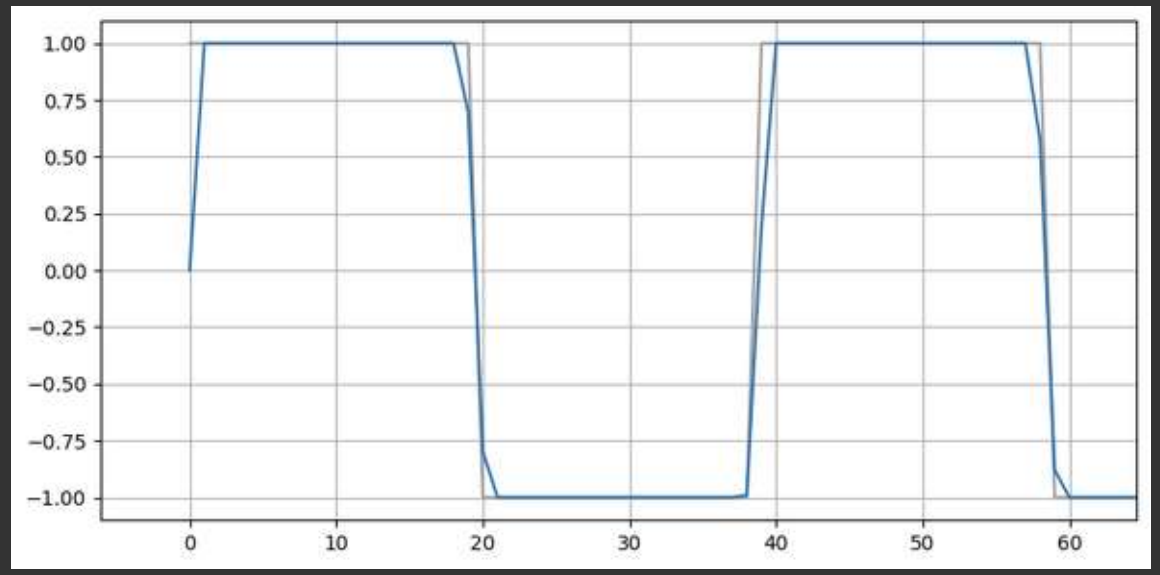
- Sygnały zapisywane w tablicy muszą być generowane w taki sposób, aby nie zawierały składowych widma powodujących aliasing.
- W przypadku sygnału prostokątnego można np. wygenerować sygnał metodą addytywną, z szeregu Fouriera, ograniczając zawartość składowych do *fs*/2.
- Sygnał traci swój kształt ze względu na brak wysokich częstotliwości.
- Istnieją metody generowania sygnałów harmonicznych o ograniczonym widmie, np. *BLIT* i *MinBLEPS*. Są one skomplikowane.

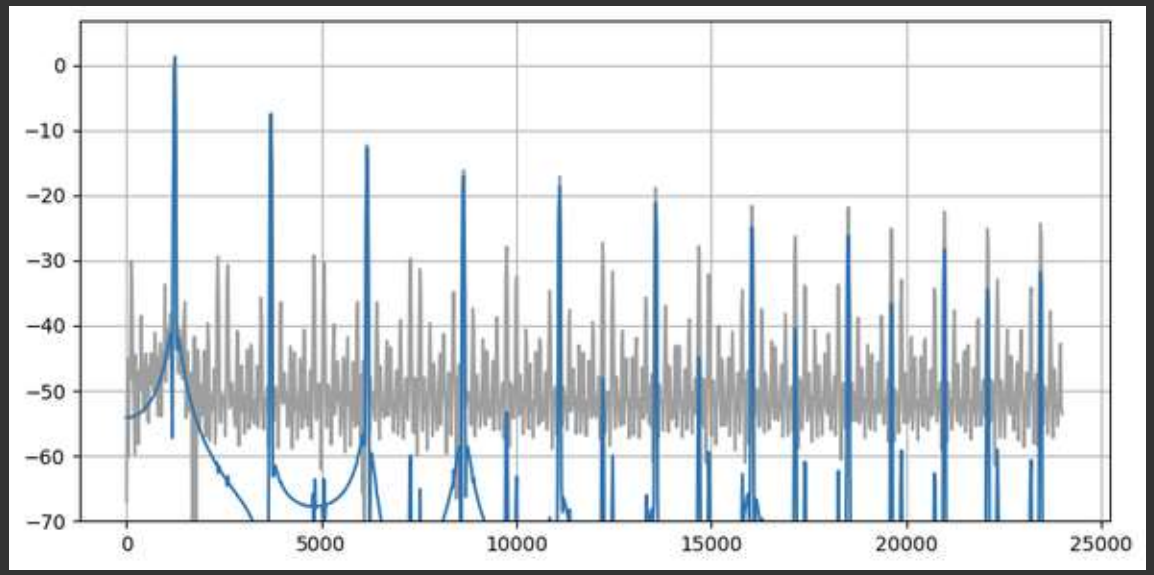


PolyBLEP

- Wysokie częstotliwości są związane z miejscami, w których fala gwałtownie zmienia swój kształt, np. „rogi” fali prostokątnej.
- PolyBLEP jest prostym algorytmem, który „przesuwa” próbki w pobliżu tych punktów za pomocą wielomianów (warunek: musimy je zlokalizować).
- Znaczna redukcja aliasingu, niewielkim kosztem obliczeniowym.

Szczegóły: https://www.kvraudio.com/forum/viewtopic.php?t=375517
https://mac.kaist.ac.kr/pubs/ValimakiPeknenNam-jasa2012.pdf





Problem zmiany wysokości dźwięku

- Musimy zmieniać częstotliwość sygnału, tak aby pokryć cały zakres wysokości dźwięku (całą klawiaturę).
- W analogowym oscylatorze wystarczy zmienić wartość napięcia sterującego.
- Przy generowaniu sygnału z pamięci, częstotliwość *f* wynika z zależności:

$$f = s \cdot fs / N$$

  - *s*: krok, z jakim przesuwamy indeks odczytu z pamięci,
  - *fs*: częstotliwość próbkowania (odczytu z pamięci), próbki na sekundę,
  - *N*: liczba próbek sygnału na jeden okres, zapisanych w pamięci.
- Np.: $N = 1024$, $fs = 48000$ Hz, $s = 1 \rightarrow f = 46{,}875$ Hz.
- Jak uzyskać inne częstotliwości dźwięku?

Problem zmiany wysokości dźwięku

- Jeszcze raz wzór:

  $f = s \cdot fs / N$

- Aby zmienić *f*, musimy zmienić przynajmniej jeden parametr po prawej stronie równania.
- Liczba próbek *N*: nie możemy jej elastycznie zmieniać, musi być całkowita.
- Częstotliwość próbkowania *fs*: można zmieniać tylko wtedy, gdy mamy przetwornik cyfrowo-analogowy oraz analogowy filtr i wzmacniacz. Stosowano takie podejście, ale jest ono mało praktyczne – wymagane wysokie częstotliwości pracy przetwornika i wysoka precyzja działania.
- Pozostaje tylko jedna możliwość: zmieniać krok odczytu *s*.

Problem zmiany wysokości dźwięku

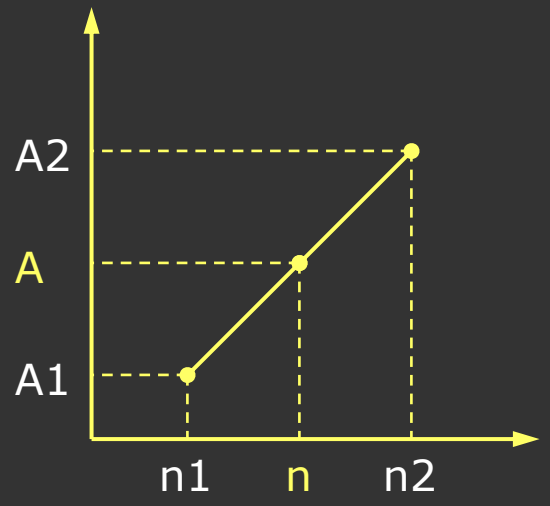
- Metoda zmiany wysokości dźwięku przez zmianę kroku odczytywania próbek z pamięci (*s*).
- Większy krok to większa częstotliwość.
- Aby uzyskać częstotliwość *f*, należy przesuwać wskaźnik odczytu o wartość: $s = f \cdot N / fs$
- np.: f = 440 Hz, N = 1024 $\rightarrow$ s = 9,386 (dla fs = 48 kHz)
  f = 1 kHz, N = 1024 $\rightarrow$ s = 21,333
- W ogólnym przypadku, krok *s* jest liczbą niecałkowitą.
- Musimy stosować interpolację, aby czytać wartości „między próbkami”.
- Interpolacja powoduje zniekształcenia sygnału – wprowadzanie dodatkowych składowych widmowych, zazwyczaj nieharmonicznych, oraz szumu.

Metody interpolacji

- Brak interpolacji (metoda „najbliższego sąsiada”) – zaokrąglanie indeksu lub obcinanie jego części ułamkowej. Najprostsza metoda, dodająca najwięcej zniekształceń do dźwięku. Stosowana np. w PPG *Wave*.
- Interpolacja liniowa – prosta i najczęściej stosowana. Mniejsze zniekształcenia.
- Dokładniejsze metody – interpolacja kubiczna (trzeciego stopnia), funkcjami *sinc*, itp. Zmniejszenie zniekształceń kosztem złożonych obliczeń. Stosowane w niektórych programowych syntezatorach.
- Zniekształcenia mają postać dodatkowych, słyszalnych składowych widma, bardzo często nieharmonicznych – psują brzmienie dźwięku.
- Im więcej próbek na jeden okres w pamięci, tym mniej zniekształceń.
- Im większa częstotliwość fali, tym więcej zniekształceń.

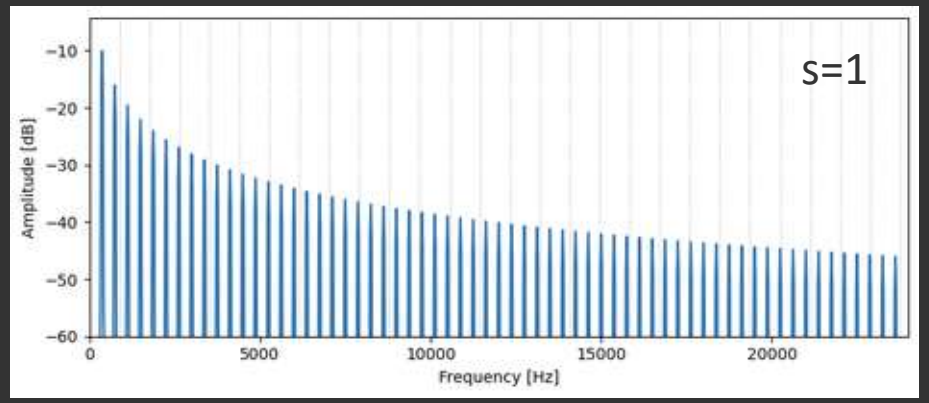
Interpolacja liniowa

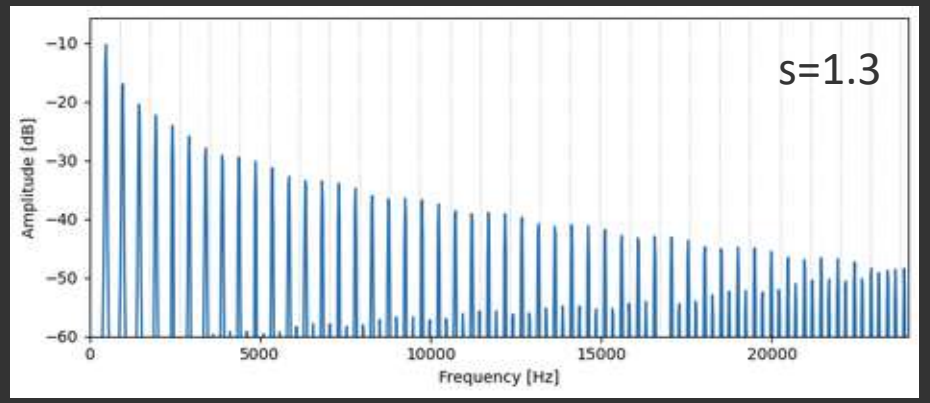
- Indeks tablicy rozbijamy na część całkowitą *c* i ułamkową *u*, np. 15,23: *c* = 15, *u* = 0,23.
- Interpolacja przez odczyt wartości na linii prostej łączącej znane próbki:
  $A = A1 + (n - n1) \cdot (A2 - A1) / (n2 - n1)$
- W naszym przypadku (*A* to wartości próbek w tablicy):
  $A[c+u] = A[c] + u \cdot (A[c+1] - A[c])$
- Praktyczna rada: warto powtórzyć pierwszą próbkę na końcu tablicy, aby uprościć interpolację między ostatnią a pierwszą próbką okresu.



Zmiana wysokości a aliasing

- Nie rozwiązaliśmy do końca problemu aliasingu widma.
- Zapisujemy w tablicy sygnał o paśmie ograniczonym do *fs*/2.
- Odczyt z krokiem *s* > 1 powoduje „rozciąganie” widma. Składowe zwiększają swoje częstotliwości i wchodzą na „alias” widma.
- Pojawia się aliasing, zwykle nieharmoniczny.
- Dla *s* > 1: składowe powyżej *fs*/2*s* spowodują aliasing.
- Odczyt z krokiem *s* < 1 powoduje powstanie pustych miejsc w widmie.

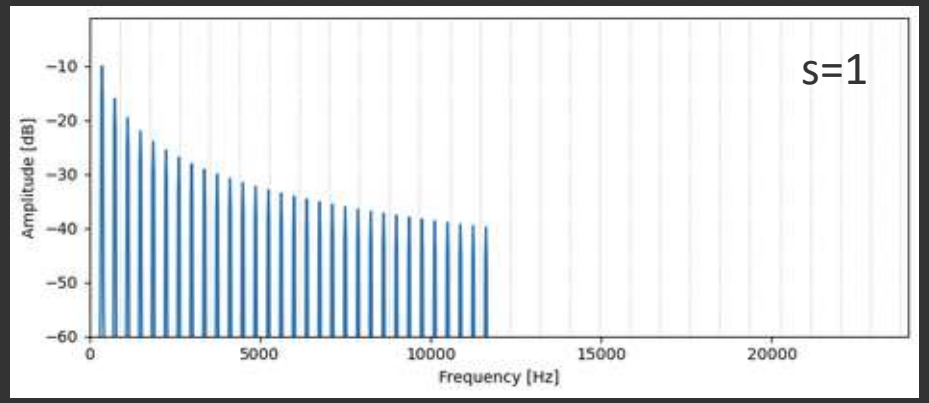


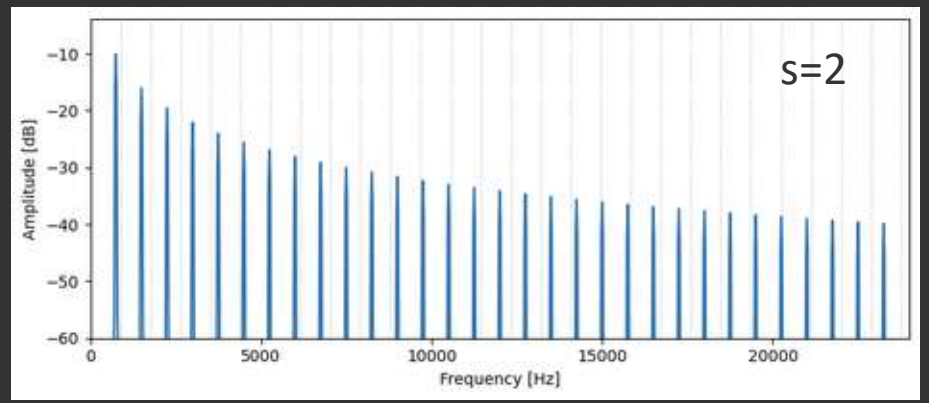


Zmiana wysokości dźwięku

Możliwe rozwiązanie problemu aliasingu przy zmianie częstotliwości.

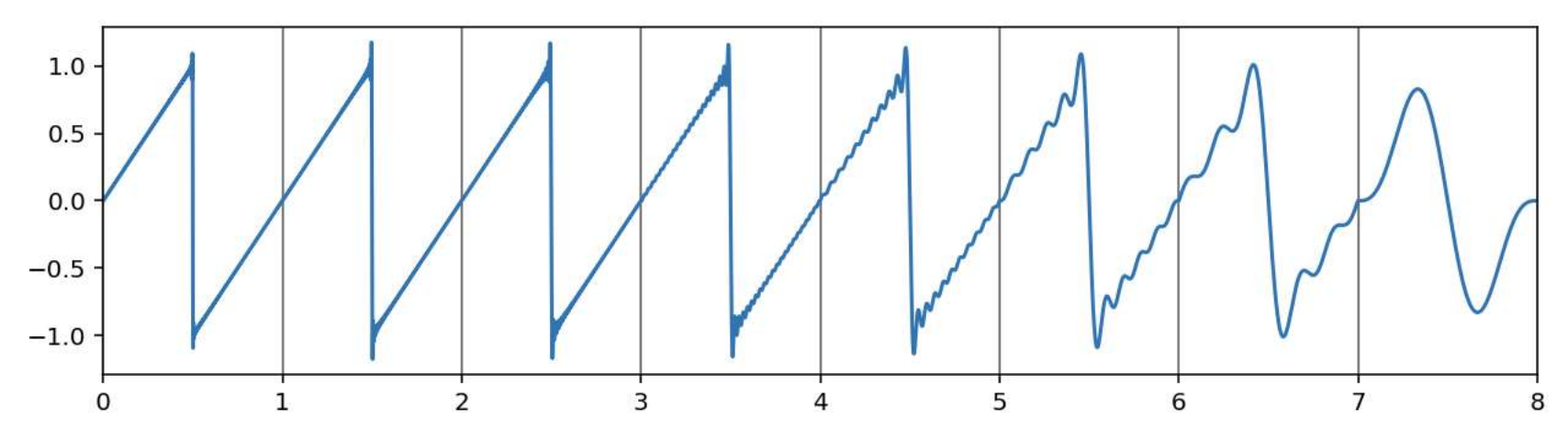
- Zakładamy zmiany *s* tylko w zakresie od 1 do 2.
- Dla *fs* = 48 kHz i *N* = 1024: *f* od 46,875 Hz do 93,75 Hz (jedna oktawa).
- Aby nie było aliasingu, ograniczamy pasmo do *fs*/4 (12 kHz).
- Wada tego podejścia: zmienna szerokość widma, a więc zmienna barwa i głośność dźwięku przy zmianie wysokości. Pasmo od *fs*/4 do *fs*/2 stopniowo się wypełnia przy zwiększaniu kroku.





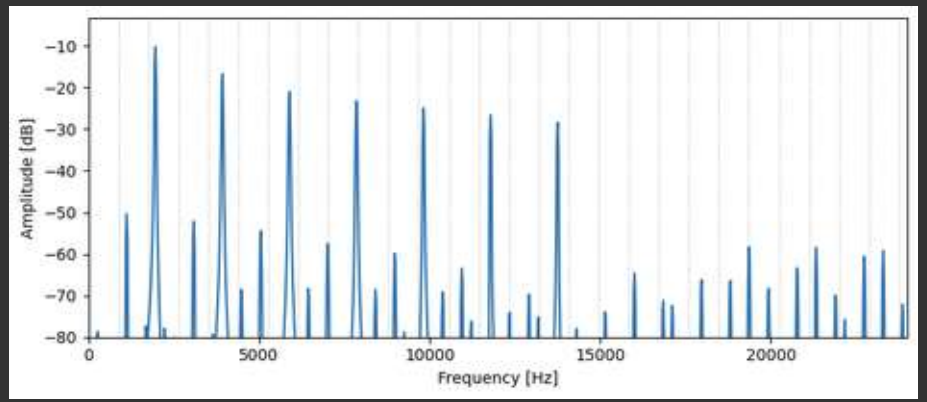
Wielopoziomowa tablica fal

- Jak uzyskać większe częstotliwości?
- Generujemy kolejny poziom tablicy: pasmo ograniczone do *fs*/8 (6 kHz), indeks *s* od 2 do 4 – to daje częstotliwości od 93,75 Hz do 187,5 Hz.
- Powtarzamy te kroki aż do pokrycia całego zakresu wysokości.
- Powstaje wielopoziomowa tablica fal.
- Przy generowaniu sygnału należy wybrać odpowiedni poziom tablicy.



Wielopoziomowa tablica fal

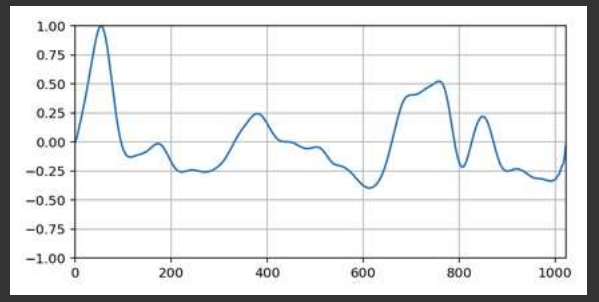
- Wydaje się, że dla każdego kolejnego poziomu tablicy można zredukować liczbę próbek o połowę i utrzymać krok *s* w zakresie od 0 do 1.
- Dokładność interpolacji liniowej zależy od liczby próbek na okres.
  Gdy próbek jest za mało, powstają niepożądane składowe widmowe.
  Przykład na wykresie: $N = 32$, $s = 1{,}31$.
- Im większa częstotliwość sygnału, tym bardziej trzeba nadpróbkować tablicę, aby zachować dokładność interpolacji i brak zniekształceń.
- W praktyce oznacza to, że trzeba zachować tę samą liczbę próbek dla wszystkich poziomów tablicy.
- Dla $N = 1024$, 8 poziomów:
  jedna fala = 32 KB pamięci
  (zapis na 4-bajtowych liczbach *float*).

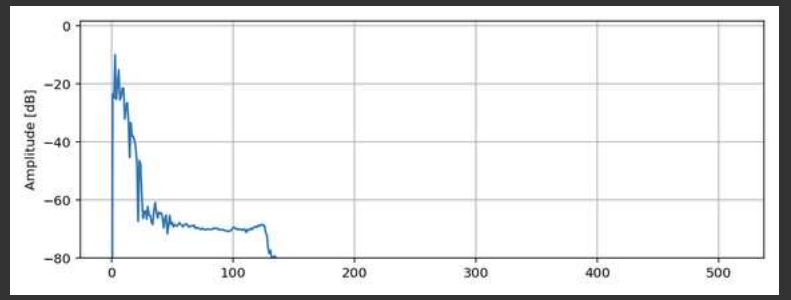


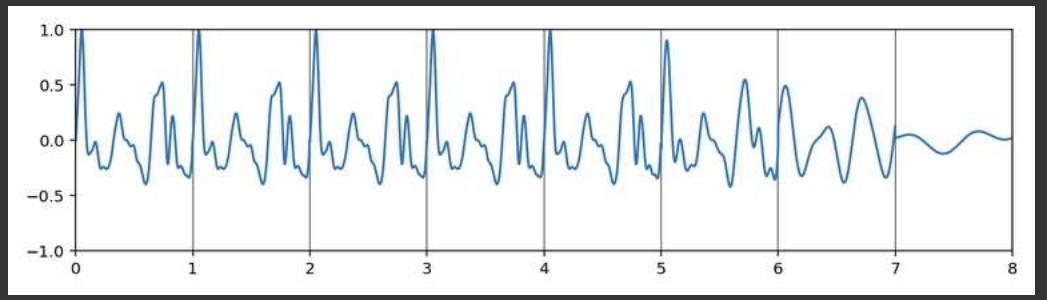
Wielopoziomowa tablica fal

Przykład generowania tablicy dla $N = 1024$.

- Bierzemy gotowy sygnał lub generujemy własny.
- Obliczamy rzeczywiste FFT o rozmiarze 1024 (513 wartości).
- Zerujemy składowe widma powyżej indeksu 256.
- Obliczamy IFFT – mamy pierwszy poziom.
- W widmie zerujemy składowe powyżej 128.
- Obliczamy IFFT – mamy drugi poziom.
- Powtarzamy, aż do uzyskania ostatniego poziomu.







Wielopoziomowa tablica fal

- Głównym problemem omawianej metody są słyszalne przejścia między poziomami: „pełne pasmo – pół pasma” – różnice brzmienia dźwięku.
- Można to obejść dopuszczając aliasing powyżej *fs*/3 (zapisując sygnał o paśmie do *fs*/3). Aliasing będzie maskowany przez silniejsze składowe, ale może być słyszalny. (Autor: Nigel Redmon*, Ear Level Engineering*).
- Można również nadpróbkować (*oversample*) sygnał, generując go z wyższą częstotliwością próbkowania (np. 96 kHz zamiast 48 kHz). Można wtedy używać pełnego pasma do 24 kHz. Aliasing zostanie odfiltrowany przy konwersji do bazowej częstotliwości 48 kHz (używa się filtrów polifazowych). Wady: dwukrotnie większe tablice, konieczność obliczania dwóch wartości na każdą próbkę i konieczność konwersji częstotliwości próbkowania.

Generowanie sygnału z tablicy

Przykład: tablica z 8 poziomami, 1024 próbek każdy, *fs* = 48 kHz.
Generujemy sygnał o częstotliwości *f* = 440 Hz.

- Obliczamy krok: *s* = 440 · 1024 / 48000 = 9,38(6).
- Znajdujemy poziom tablicy – poziom 4 (*s* od 8 do 16, *f* od 375 Hz do 750 Hz).
- Ustawiamy indeks odczytu na 0, wysyłamy próbkę o indeksie 0 na wyjście.
- Zwiększamy indeks do 9,38(6). Interpolujemy liniowo między próbkami 9 i 10 w tablicy na poziomie 4. Wysyłamy wynik na wyjście.
- Zwiększamy indeks o 9,38(6), interpolujemy, powtarzamy.
- Gdy indeks przekroczy długość tablicy (1024), „zawijamy” go.
  Np. 1032,5(3) → 8,5(3).
- Jeżeli zmieni się *f* – przeliczamy krok, ew. znajdujemy nowy poziom tablicy.

Rozwój syntezy tablicowej

- Początkowo termin „synteza tablicowa” (*wavetable synthesis*) odnosił się do tablicy fal – pojedynczych okresów (np. *PPG Wave*).
- Z czasem dostępne stały się większe i tańsze pamięci.
- Termin *wavetable* zaczął być stosowany w odniesieniu do wszystkich metod generujących dźwięk poprzez odczyt tablicy próbek sygnału, nie „tablicy fal”.
- Czasami nawet używano tej nazwy równoważnie z samplingiem.
- Termin „synteza tablicowa” stał się nieprecyzyjny.
- Próba sprecyzowania:
  - synteza tablicowa: pojedyncze okresy fal w tablicy, filtracja, przetwarzanie;
  - synteza „samplingowa”, oparta na próbkach dźwięku: cały dźwięk odczytywany z pamięci, zapętlanie wybranego (dłuższego) fragmentu.

Synteza tablicowa - podsumowanie

Zalety w stosunku do metody subtraktywnej:

- większa różnorodność sygnałów dostępnych w generatorze cyfrowym,
- większa możliwość kształtowania barwy dźwięku w generatorze,
- większe możliwości brzmieniowe (szersza gama brzmień),
- stabilność wysokości dźwięku.

Wady:

- ograniczenia pamięci – tylko krótkie sygnały,
- problematyczna zmiana wysokości dźwięku,
- problem aliasingu,
- bardziej cyfrowe, ostre brzmienie niż w analogowych instrumentach.

Literatura

- Wikipedia (wersja angielska)
- Dokumentacja syntezatora PPG Wave: http://www.hermannseib.com/english/synths/ppg/docs.htm
- Dokumentacja instrumentu Waldorf Wave 3.V (VST): http://tiny.pl/glhnm
- Program PPG Wave Simulator (VST): http://www.hermannseib.com/english/synths/ppg/wavesim.htm
- Ear Level: Wavetable oscillator. http://www.earlevel.com/main/category/digital-audio/oscillators/wavetable-oscillators/
- Tim Stilson, Julius Smith: *Alias-free digital synthesis of classic analog waveforms* (BLIT). https://ccrma.stanford.edu/~stilti/papers/blit.pdf
- Eli Brandt: *Hard sync without aliasing* (MinBLEPs). https://www.cs.cmu.edu/~eli/papers/icmc01-hardsync.pdf
- Vesa Valimaki, Jussi Pekonen, Juhan Nam: *Perceptually informed synthesis of bandlimited classical waveforms using integrated polynomial interpolation*. JASA 131(1), 2012. https://mac.kaist.ac.kr/pubs/ValimakiPeknenNam-jasa2012.pdf