Independent Project EE3025

GOUTHAM.AGV , EE17BTECH11001

29 Dec 2020

Chapter 1

Introduction to Eda Playground

1.1 Blink Program

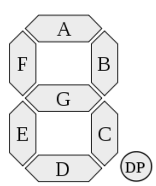
Implementation of Blink program on Eda Playground(online simulator) can be found here and GitHub link here
.

A short video explanation of working with the simulator and code can be found here.

1.2 Seven Segment display counter

Implementation of Seven segment display counter on Eda Playground can be found here and GithuB link here
.

A short video explanation of working with the simulator and code can be found here.



Chapter 2

FFT Algorithm

2.1 Background

Let x[n] be a series of complex signals, for 0≤n≤N-1 and X[k] the Discrete Fourier Transform of 0≤k≤N-1. The frequency domain signals X[k] can be obtained according to *equation 2.2* or in its matrix form as can be seen in *equation 2.3*, where

$$W_N^k n = exp(\frac{2ikn\pi}{N}) \tag{2.1}$$

$$X[k] = \sum_{n=0}^{N-1} X[n] W_N^k n \tag{2.2}$$

$$\begin{bmatrix} X[0] \\ \vdots \\ X[k] \end{bmatrix} = \begin{bmatrix} W_N^{0*0} & \dots & W_N^{0*N-1} \\ \vdots & \ddots & \vdots \\ W_N^{n-1*0} & \dots & w_N^{N-1*N-1} \end{bmatrix} * \begin{bmatrix} x[0] \\ \vdots \\ x[k] \end{bmatrix} \tag{2.3}$$

The computation of X[k]has complexity of 0(n^2). However, the expression of the equation 2.2 may be split in two terms according to equation 2.4.

$$X[k] = \sum_{n=0}^{N-1} X[2n] W_N^{2kn} + \sum_{n=0}^{N-1} X[2n+1] W_N^{2kn+1} \tag{2.4}$$

Note that applying the properties $W_N^{2kn} = W_{N/2}^{kn}$ and $W_N^{k+N/2} = -W_N^{kn}$ in the above equation, we get

$$\begin{aligned} X[K] &= Xe[k] + W_N^k Xo[k] \\ X[K+N/2] &= Xe[k] - W_N^k Xo[k] \end{aligned} \tag{2.5}$$

where

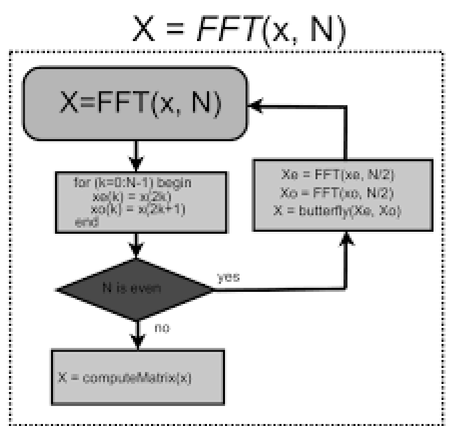
$$Xe[k] = \sum_{n=0}^{\frac{N}{2}-1} X[n] W_{\frac{N}{2}}^{kn}$$

$$Xo[k] = \sum_{n=0}^{\frac{N}{2}-1} X[2n+1] W_{\frac{N}{2}}^{kn}$$

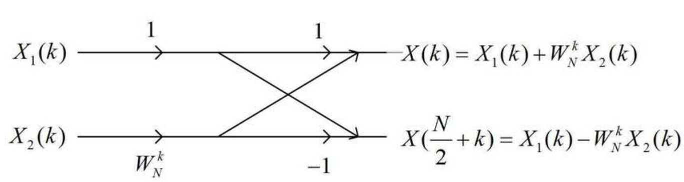
According to equation 2.5, if N is a power of 2, the computation of X[k] has complexity of $0(N*\log_2(N))$.

2.2 Recursive Algorithm

A very simple algorithm to compute the FFT can be defined taking advantage of the recursive nature of the FFT, as can be seen in



if the size N of the FFT is even then call two FFT of order N/2, one to compute the Fourier Transform of the signals with even index (x[2n]) and other to compute the signals with odd index (x[2n+1]).
The Fourier Transform will be scaled with the twiddle factor $W_N{}^k$. if N is odd



then the FFT will be slowly calculated using the Fourier matrix of equation 2.3(Matrix multiplication).

If the length N of the FFT is of the form N= m * 2^{p}, the the complexity of the algorithm of Figure 1 will be $O(\mathrm{m}^2)\times O(\mathrm{P} * \log_2(\mathrm{p}))$. Sudo code of the above algorithm considering genenral case m=1 is

FFT Algorithm

NOTE: Procedure FFT is presented here in pseudo-code,
for a generic field F in which it is possible to define ω,
a primitive n-th root of unity.

```
procedure FFT (A, n, w)

    # Preconditions:
    #   A is a Vector of length n;
    #   n is a power of 2;
    #   w is a primitive n-th root of unity.
    #
    # The Vector A represents the polynomial
    #   a(z)  =  A[1] + A[2]*z + ... + A[n]*z^(n-1) .
    #
    # The value returned is a Vector of the values
    #    [ a(1), a(w), a(w^2), ... , a(w^(n-1)) ]
    # computed via a recursive FFT algorithm.

    if n = 1 then
        return A
    else
        Aeven <-- Vector( [A[1], A[3], ..., A[n-1]] )
        Aodd  <-- Vector( [A[2], A[4], ..., A[n]] )

        Veven <-- FFT( Aeven, n/2, w^2 )
        Vodd <-- FFT( Aodd, n/2, w^2 )

        V <-- Vector(n)  # Define a Vector of length n
        for i from 1 to n/2 do
            V[i] <-- Veven[i] + w^(i-1)*Vodd[i]
            V[n/2 + i] <-- Veven[i] - w^(i-1)*Vodd[i]
        end do
        return V
    end if
end procedure
```

2.3 Code and Links

SystemVerilog code implemented in Eda Playground can be found here and corresponding GitHub link is here.
A short video explaining the recursive fft code using SystemVerilog can be found here.
A faster algorithm iterative FFT is implemented in c code; Github link for c is here.
Github Link for all the codes in project is here.