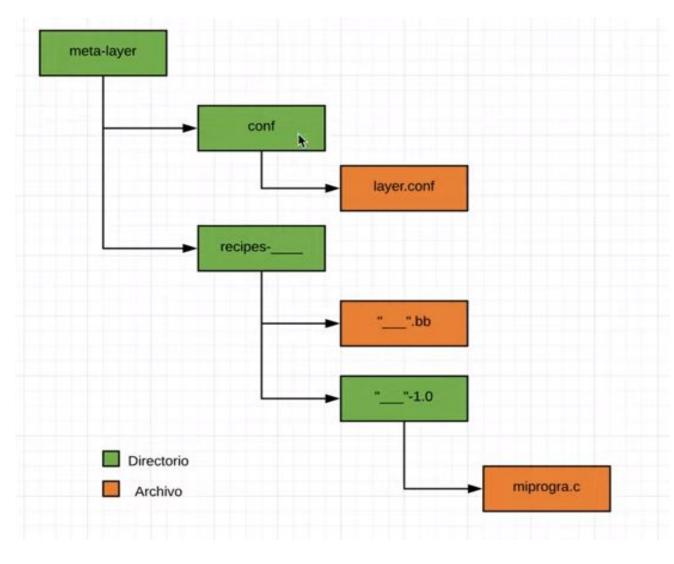
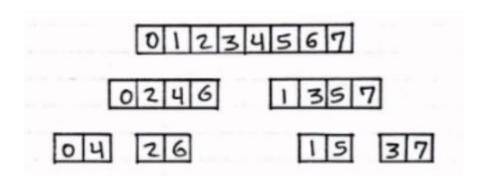
Diagrama básico para Yocto



Para ensamblador:

Se adjunta Nasm a meta/conf/bitbake.conf

Para cálculo



$$W_N^K=e^{-irac{2\pi}{N}k}$$

Primera parte (Generar función)

Pedir usuario de que tamaño quiere hacer la función: tamaño "n"

Crear funciones auxiliares (Generador de números random) Random

Inicio de ciclo para generar función

Llamar función Random, generar numero

Colocar dichos números en una lista tamaño "n"

Fin de ciclo para generar función (Lista llena)

Imprime esa función.

Primera parte (Generar función) NASM

Hacer espacios de memoria para colocar la función, ejemplo video 8 en dirección 10 y la siguientes direcciones apuntan a cada valor de la función. Espacio de memoria para guardar el numero random

Para esto se utiliza el código del video 4 del tutorial de NASM

Se busco en internet como utilizar el clock interno y generar números random a partir de el y lo coloca en una subrutina como en el video 6

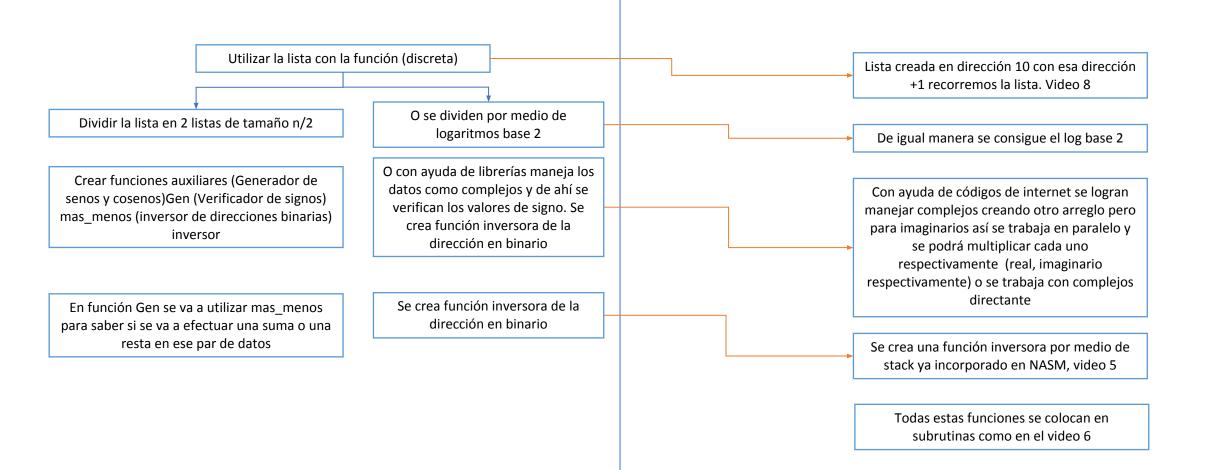
Se genera un loop con el código del video 3, con ayuda de jumps repitiendo la función hasta llegar a cierta condición

Se hace llamado de la subrutina random para generar un numero dentro del loop y colocarla en el espacio de memoria asignado previamente (con +1 cada vez que se repita el ciclo)

Fin del ciclo, de recorrio hasta llegar al tamaño "n"

Se hace uso del código en el video 8 del tutorial de NASM, donde imprimimos lo que esta en la dirección 10, que en este caso seria la serie de números random completa Segunda parte (Preparar para hacer FFT)librerías

Segunda parte (Preparar para hacer FFT) NASM



Tercera parte (hacer FFT en función e imprimir resultado) NASM Tercera parte (hacer FFT en función e imprimir resultado) Crea espacios de memorias para colocar Inicio de ciclo para generar FFT inversor, multiplicación de la función y guardado de dirección de la función principal Se genera un loop con el código del video 3, Llama el dato y utiliza "inversor" para generar con ayuda de jumps repitiendo la función su pareja y los guarda hasta llegar a cierta condición Coloca dirección +1 para caer al primer dato y Coloca el dato y su dato en dirección inversor por medio de su dirección hace la subrutina en "Gen" y lo guarda inversión y encuentra su pareja y la guarda en inverso Coloca el dato en la dirección que estamos Este resultado lo multiplica por el primer dato trabajando y inverso en la función para y sobre escribe en la dirección del primer efectuar la subrutina de senos y cosenos y lo dato. Lo mismo ocurre con el dato invertido guarda en multiplicación de funcion Este ultimo dato lo guarda en la función Fin de ciclo para generar FFT (Utilizo toda la principal y sigue el ciclo primera mitad de la lista) Final del ciclo Imprime resultado de la FFT sobre la función Se hace uso del código en el video 8 del generada random. tutorial de NASM, donde imprimimos lo que esta en la dirección 10, que en este caso seria la función ya con la transformada de la FFT

Por complejidad primero se programo en C++ (programa ya conocido) y de ahí se paso a la lógica de NASM

Dificultades al implementar en NASM

- Números flotantes, se usa enteros
- Necesidad de una lista
- Saber el tamaño de las funciones y crear suficientes espacios de memoria
- Trabajar números complejos
- Trabajar senos y cosenos
- Trabajar logaritmos
- Manejo de registros para modificar una memoria

Soluciones al implementar en NASM

- Uso de compiladores para pasar código de C++/C a NASM
- Manejar la misma lógica de resolución del problema