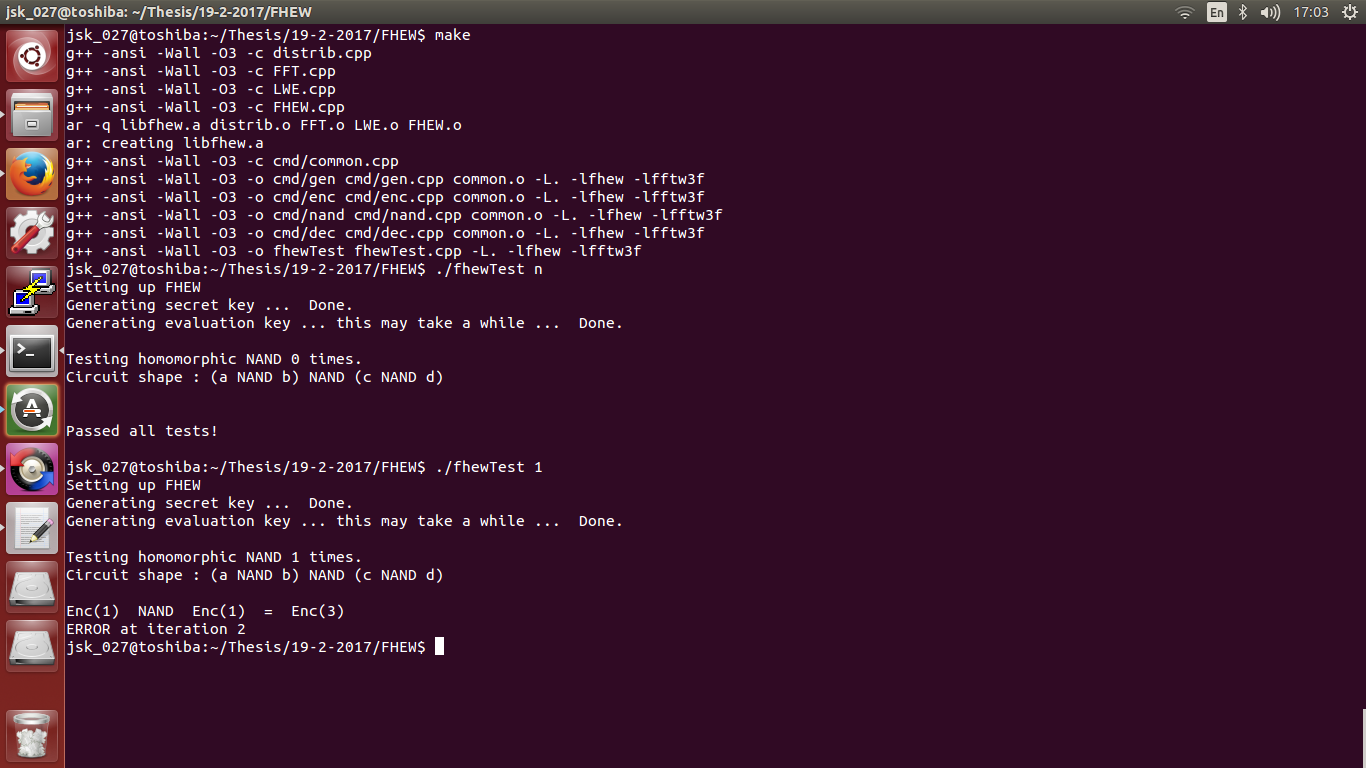
<http://www.fftw.org/fftw3_doc/Installation-on-Unix.html#Installation-on-Unix>

<http://www.fftw.org/fftw3_doc/Precision.html#Precision>

* Link to the single/long-double libraries; on Unix, -lfftw3f or -lfftw3l instead of (or in addition to) -lfftw3.
* Include the *same* <fftw3.h> header file.
* Replace all lowercase instances of ‘fftw\_’ with ‘fftwf\_’ or ‘fftwl\_’ for single or long-double precision, respectively. (fftw\_complex becomes fftwf\_complex, fftw\_execute becomes fftwf\_execute, etcetera.)
* Uppercase names, i.e. names beginning with ‘FFTW\_’, remain the same.
* Replace double with float or long double for subroutine parameters.



<https://github.com/jtfell/c-fft> - Accuracy is preserved.

* Integrate this FFT with the software. => Working (Accuracy preserved – only for double precision floating point)



* **Not synthesizable in Vivado – make It synthesizable.**
* Remove function return types (void type),
* Remove dynamic memory allocations,
* 2D Array to 1D Array
* **Replace sin () and cos () functions with equivalent look up tables** => Precision Lost



**Revert Look-up table changes for now!**

* Apply Directives and find the most optimal solution.



**FFT Butterfly**

* **Tested to be optimal and produces same accuracy as Cooley Tuckey – improved latency**

**Vivado HLS FFT IP Core**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attributes** | **Cooley Tuckey FFT** | **Butterfly FFT** | **FFT IP CORE (Xilinx)** |
| Best suited directives | C:\Users\Swarna\AppData\Local\Microsoft\Windows\INetCache\Content.Word\DirectivesForCT.PNG | C:\Users\Swarna\AppData\Local\Microsoft\Windows\INetCache\Content.Word\DirectivesAppliedForButterfly.png  Without specifying the loop trip count explicitly for the loops 1 and 2, there is no way for the tool to determine the number of iterations of a variable length loop at runtime. For High-level synthesis, all values need to be statically defined, to get an estimate of Latency. | Not Applicable.    The IP Core in the design is fed with data in hexadecimal format and software verification is performed by comparing the results with equivalent double precision floating point values.  As IP Core is a hard block, there is no scope for further optimization using this approach. Should explore how it can be used for our purpose. (**Reference Documentation:** <https://www.xilinx.com/support/documentation/ip_documentation/ds808_xfft.pdf>) |
| Execution time with Best case (min) latency from synthesis report | 5387290 x 11.96 ns = **0.064 seconds** | Time = No. of clock cycles x Clock period = 1210 x 8.23 ns = **9.95 microseconds** | **0.31 microseconds** |
| Execution time with worst case (max) latency from synthesis report | 5387290 x 11.96 ns = **0.064 seconds** | Time = No. of clock cycles x Clock period = 49871994 x 8.23 ns = **0.41 seconds** | **0.31 microseconds** |

As Virtex 7 FPGA is not available in Vivado HLS, I have done the FFTs using the board, **Virtex 6 ML605 Evaluation Platform**. Please find the latency results for the same attached.

Only the most beneficial directives identified from previous experiments have been used to get the results on Virtex 6 FPGA.

**Cooley Tuckey Butterfly**

|  |  |  |
| --- | --- | --- |
| **Attributes** | **Cooley Tuckey FFT** | **Butterfly FFT** |
| Execution time with Best case (min) latency from synthesis report | Time = No. of cycles x duration of one clock pulse = 4349954 x 12.48 ns = **0.05 seconds** | Time = No. of cycles x duration of one clock pulse = 1190 x 9.5 ns = **0.11305 microseconds** |
| Execution time with worst case (max) latency from synthesis report | Time = No. of cycles x duration of one clock pulse = 4349954 x 12.48 ns = **0.05 seconds**  **(Latency is more stable)** | Time = No. of cycles x duration of one clock pulse = 41997414 x 9.5 ns = **0.398 seconds**  (Latency changes drastically for best and worst cases) |