Silicon Photonics Course Design Proposal

Eduardo Barrera Ramirez, *edX Course 2025*

***Abstract*—In this report I aim to explain the design, simulation, and measurement analysis of a set of Silicon photonic circuits, which include Mach-Zehnder interferometers, a lattice filter, and an adjustable splitter. The objective of this work is to fabricate these structures and compare their experimental response to their simulation results. In particular, I intend to extract the free spectral range (FSR) from each MZI device and determine the respective group index.**

I. Introduction

THE design work outlined in this report is separated into three difference type of devices: MZI, adjustable splitter, and a lattice filter. The design proposal has a layout file called EBeam_ebramire.gds

April 30, 2025

II. Waveguide Simulations

All of the structures included in the design proposal utilize the same waveguide geometry, polarization, and a bend radius of $5\mu m$ during the layout process. The chosen waveguide geometry was the standard 220 nm tall and 500 nm wide rectangular strip waveguide. The TE polarization was also chosen for all of the structures in this design. The electric field intensity for the TE mode of this strip waveguide is shown in Fig.1. Furthermore, a plot of Lumerical Mode simulations for the effective index (n_{eff}) and the group index (n_g) as a function of wavelength (λ) for the TE mode is shown in Fig.2. A Taylor series expansion, shown in Eq.1, was used to fit the simulation data for n_{eff} vs. λ, which yielded the fit parameters shown in TableI. We can determine the group index n_g and dispersion D by using their definitions from Eqs.2 and 3 and the taylor expansion of n_{eff}. In order to compare the lumerical simulations for the group index with the experimental data, Fig.2 includes the group index extracted from the experimental response (blue) of a Mach-Zehnder Interferometer (MZI4). This extracted group index was calculated using Eq.2 and using fit parameters similar to those shown in Table I. For more details on the design of MZI4 and the methods used to extract the group index, see Section VIII.

$$n_{eff} = n_1 + n_2(\lambda - \lambda_o) + n_3(\lambda - \lambda_o)^2 \tag{1}$$

$$n_g(\lambda) = n_{eff} - \lambda\frac{dn_{eff}}{d\lambda} = n_1 - \lambda_o n_2 + (\lambda_o^2 - \lambda^2)n_3 \tag{2}$$

$$D = \frac{-2\lambda_o n_3}{c} \tag{3}$$

Report submitted Fab 3, 2025;

Fit Parameter	Value
n_1	2.44487166
n_2	-1.13050187 μm^{-1}
n_3	-0.04209636 μm^{-2}
n_g	4.197
D	4.353x10^{-4} s/m^2

TABLE I: Fitting parameters for compact model of the strip waveguide

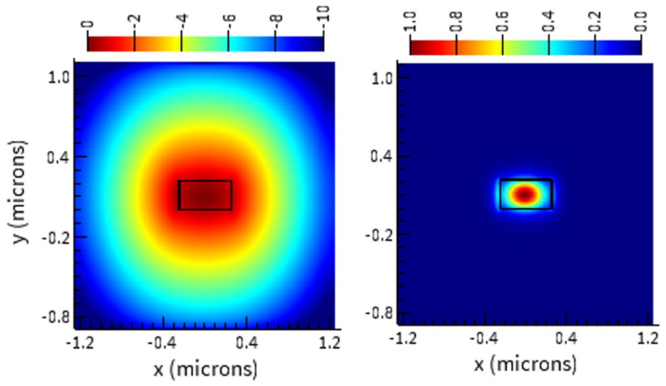

Fig. 1: Lumerical Mode simulations of the waveguide TE mode in log (left) and linear scale (right).

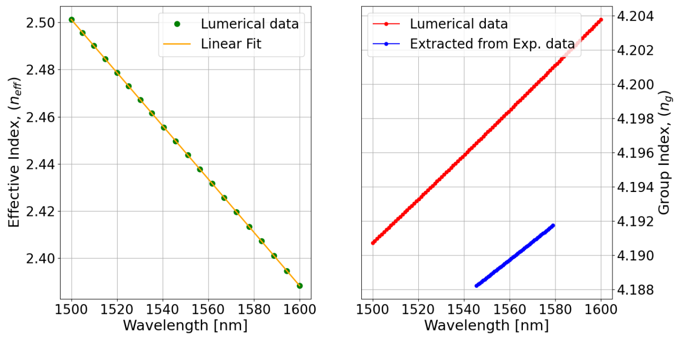

Fig. 2: Plot of the effective index (left) and group index (right) as a function of wavelength. The group index data extracted from experimental data belongs to MZI4.

III. Mach-Zehnder Interferometer

The layout includes a total of seven different MZIs in which the path length difference (ΔL) ranged from $100\mu m$ to $14150\mu m$. To investigate the impact of fabrication variability on MZI performance, five MZIs with a constant ΔL of $1600\mu m$ were added. These identical MZIs were labeled as MZI_TE_1600umFabVar1 to MZI_TE_1600umFabVar4 and can help us infer the intra-chip variability.

$$\Delta\lambda = \frac{\lambda^2}{n_g \Delta L} \tag{4}$$

When simulating the MZI circuit model, we used a waveguide loss in dB/cm was 5.9x10^{-4} dB/cm. In order to determine the "Expected FSR", Eq.4 was used to calculate the FSR for each path length difference, where n_g was estimated to be 4.197 from Fig.2. A circuit model for each MZI was created and simulated in Lumerical INTERCONNECT, resulting in an output response as a function of wavelength. The simulated response for MZI4 is shown in Fig3 overlaid with the experimentally measured response. One can see that the grating couplers have a significantly different performance. This experimental data will be fit to the MZI transfer function in section VIII. The Lumerical INTERCONNECT simulations allowed the extraction of the FSR value near λ equal to 1550nm, referred to as "Simulated FSR". Table II summarizes the ΔL, Expected FSR, and Simulated FSR for each MZI structure included in this design.

MZI ID	ΔL [μm]	Expected FSR [nm]	Simulated FSR [nm]
MZI_TE_100um	100	5.7238	5.7031
MZI_TE_200um	200	2.8619	2.8621
MZI_TE_400um	400	1.4310	1.4285
MZI_TE_800um	800	0.7155	0.7149
MZI_TE_1600um	1600	0.3577	0.3551
MZI_TE_3200um	3200	0.1789	0.1751
MZI_TE_14150um	14150	0.0405	0.0400

TABLE II: Path length difference and FSR values for the MZI devices

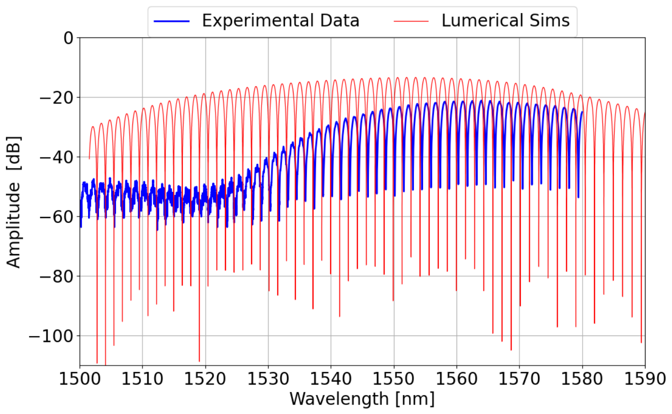

Fig. 3: Lumerical simulation and Experimental data for the transmission spectrum (gain) for MZI4 mentioned in Table II

IV. Adjustable Splitter

The design for the adjustable splitter is based on the transfer function of the interferometer, shown in Eq.5, which can be rearranged to solve for ΔL, refer to Eq.6. The adjustable splitter was labelled as "Splitter_TE_25-75" and chosen to have a ratio of 25/75, which yields a ΔL of 105.4nm by setting $\frac{I_o}{I_i}$ equal to 0.75. Fig.4 shows the two outputs from the adjustable splitter which confirms that the output "Out1" and "Out2" are roughly 75.82% and 24.18%, respectively, at about 1550nm.

$$\frac{I_o}{I_i} = \frac{1}{2}\left[1 + cos\left(\frac{2\pi n_{eff}}{\lambda_o}\Delta L\right)\right] = \frac{1}{2}(1 + cos(\beta\Delta L)) \tag{5}$$

$$\Delta L = \frac{\lambda cos^{-1}\left(2\frac{I_o}{I_i} - 1\right)}{2\pi n_g} \tag{6}$$

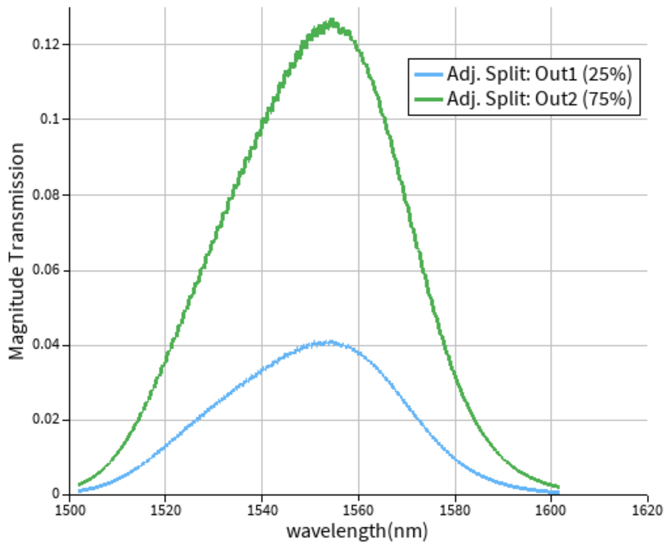

Fig. 4: Magnitude of the transmission outputs of a 25%-75% adjustable splitter

V. Lattice Filter: Chained MZIs

The lattice filter included in the design proposal is based on three MZIs that create a four channel demultiplexing. There are two copies of the same lattice filter, labelled as "DeMux1_TE_1560-1545" and "DeMux2_TE_1560-1545", because there are not enough outputs to measure the four outputs of the lattice filter. Therefore, I made a copy ("DeMux2_TE_1560-1545") where I connected the missing output from "DeMux1_TE_1560-1545".

I used broadband directional couplers to build each MZI, which had two outputs with a phase difference of π. I wanted to have a channel separation ($\delta\lambda$) of 5nm for my demultiplexed channels. Since there is a π phase difference between the output of my first MZI, I chose a path length difference for this first MZI (MZI-1) such that I would obtain a FSR equal to 10nm. This FSR choice is more evident when one sees the two outputs of MZI-1, shown in Fig5. Based on Eq7, ΔL was estimated to be 57.24μm. I slightly modified ΔL to 57.52μm in order to center the oscillations of the two outputs at 1550 and 1555, respectively, as shown in Fig5. In this figure, it is clear that if we consider the two outputs there are peaks separated by about 5nm and centered on 1550nm.

$$\Delta L = \frac{\lambda^2}{\delta\lambda n_g} \tag{7}$$

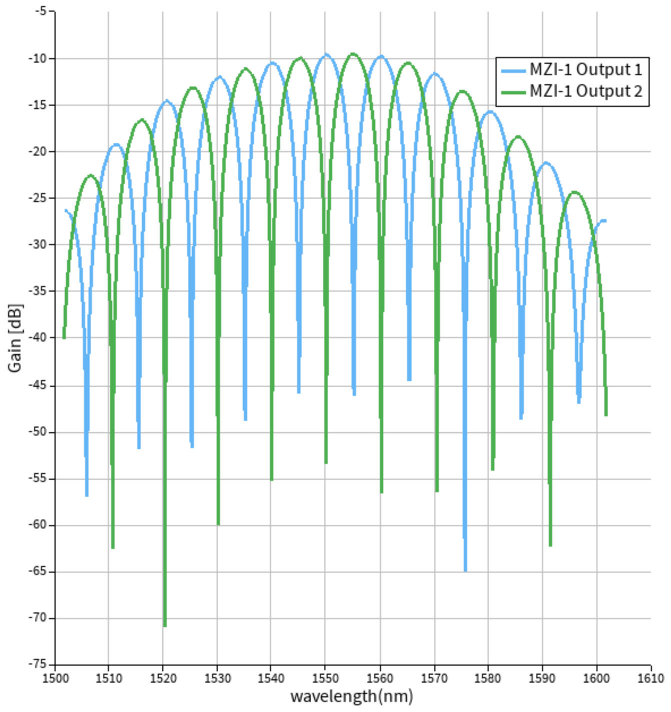

Fig. 5: Two outputs for the first MZI in the lattice filter

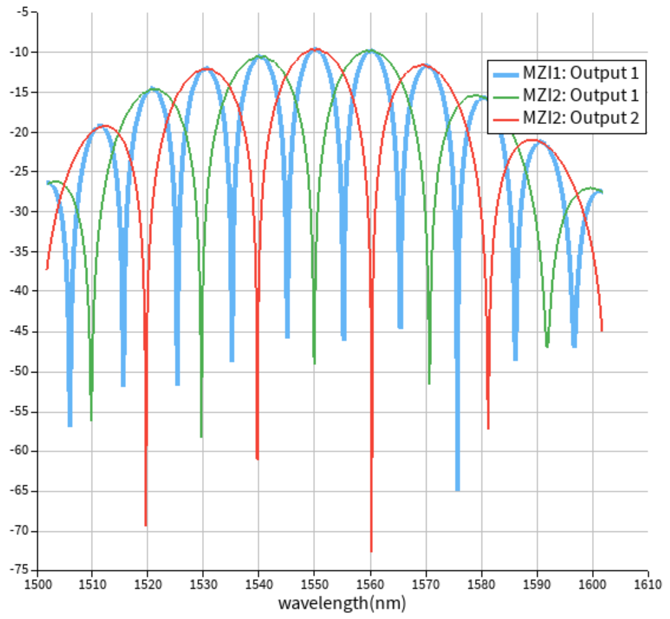

Fig. 6: Comparison between one output from MZI-1 and the two outputs of MZI-2a

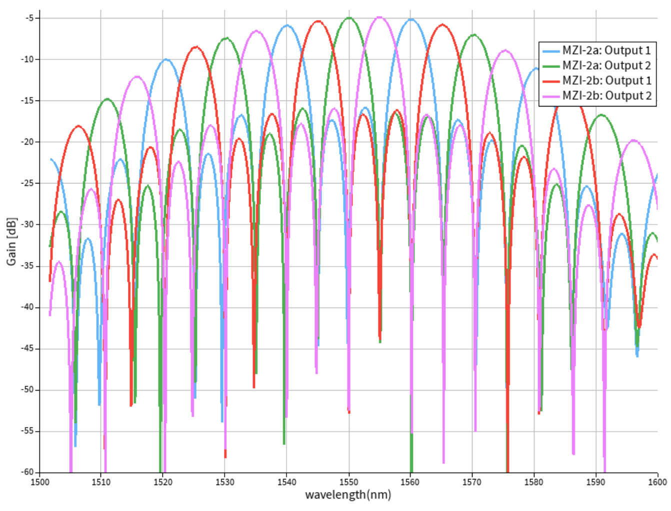

Fig. 7: Final spectrum of the lattice filter

Next, my approach was to add two more MZIs and feed each outputs from MZI-1 into these second MZIs (MZI-2a and MZI-2b). Then, the task would be to set the FSR of the MZI-2a and MZI-2b to be equal to $2\delta\lambda$ and centered at 1550nm or 1555nm, respectively. Fig6 shows one of the outputs of MZI-1 and the two outputs for MZI-2a without connect the two MZIs. One can see that by doubling the FSR of MZI-2a with respect to MZI-1 and properly aligning the two transmission spectra, we can select a set of peaks to be transmitted into each of the two outputs of MZI-2a. The same exercise applied for the second output of MZI-1 and the second MZI-2b. Once again, I used Eq7 to calculate ΔL for MZI-2a and MZI-2b, using $2\delta\lambda$ instead of $\delta\lambda$. After making fine phase adjustments to align the spectra, we obtain the overall transmission spectrum shown in Fig7 after connect MZI-1, MZI-2a, and MZI-2b. Fig8 shows a reduced wavelength range, where one can see the four individual channels starting at 1545nm to 1560nm. It is evident that the reponse of this lattice filter does not have transmission peaks with flat peaks, which is one main drawback of this implementation.

VI. Fabrication

The photonic devices were fabricated using the NanoSOI MPW fabrication process by Applied Nanotools Inc. (http://www.appliednt.com/nanosoi; Edmonton, Canada) which is based on direct-write 100 keV electron beam lithography technology. Silicon-on-insulator wafers of 200 mm diameter, 220 nm device thickness and 2 μ m buffer oxide thickness are used as the base material for the fabrication. The wafer was pre-diced into square substrates with dimensions of 25x25 mm, and lines were scribed into the substrate backsides to facilitate easy separation into smaller chips once fabrication was complete. After an initial wafer clean using piranha solution (3:1 H_2SO_4:H_2O_2) for 15 minutes and water/IPA rinse, hydrogen silsesquioxane (HSQ) resist was spin-coated onto the substrate and heated to evaporate the solvent. The photonic devices were patterned using a JEOL JBX-8100FS electron beam instrument at The University of British Columbia. The exposure dosage of the design was corrected for proximity effects that result from the backscatter of electrons from exposure of nearby features. Shape writing order was optimized for efficient patterning and minimal beam drift. After the e-beam exposure and subsequent development with a tetramethylammonium sulfate (TMAH) solution, the devices were inspected optically for residues and/or defects. The chips were then mounted on a 4” handle

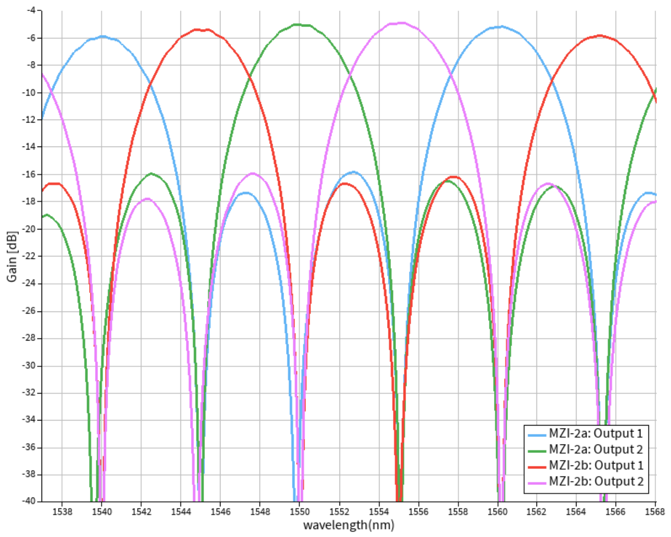

Fig. 8: Zoomed in view of the final spectrum of the lattice filter

wafer and underwent an anisotropic ICP-RIE etch process using chlorine after qualification of the etch rate. The resist was removed from the surface of the devices using a 10:1 buffer oxide wet etch, and the devices were inspected using a scanning electron microscope (SEM) to verify patterning and etch quality. A 2.2 μm oxide cladding was deposited using a plasma-enhanced chemical vapour deposition (PECVD) process based on tetraethyl orthosilicate (TEOS) at 300^{o}C. Reflectrometry measurements were performed throughout the process to verify the device layer, buffer oxide and cladding thicknesses before delivery.

A. *Fabrication Variability: Corner Analysis*

The final dimensions of fabricated photonic structures are expected to have deviations from their intended design values due to manufacturing variability. This variability has a contribution from alterations in the shape and dimensions of the structures during fabrication. Factors such as the proximity effect in e-beam lithography or the sidewall profile after dry-etching can lead to modifications in the shape of a targeted rectangular waveguide into a trapezoidal-shaped waveguide with smooth corners. This is also known as the smoothing effect. Moreover, the wafer material has inherent variability in the device Si layer thickness which has a direct impact on the height of the strip waveguide. In an effort to assess the fabrication variability, a corner analysis was performed, requiring the simulation of the strip waveguide model for four different "corner" cases. These cases vary the waveguide height and width from 215.3 nm to 223.1 nm and 470 nm to 510 nm, respectively. These ranges were suggested as reasonable based on the historic fabrication data using the Applied Nanotools process. Section VIII will discuss the extraction of the group index from experimental data which happened to be lower than the minimum group index value from this corner analysis. Therefore, the waveguide height and width was increased since this tends to lower the group index. The waveguide height was constrained to a minimum of 207.5 nm, which corresponds to a 6σ lower bound on the device's silicon layer thickness [1]. Meanwhile the width was increased to 535 nm in order to include the simulated group index value. Every corner case considered was assigned a label according to Table III, where the label target represents the intended design dimensions.

Label Width, Height	Width [nm]	Height [nm]
H, H	510	223.5
H, L	510	215.3
L, H	470	223.5
L, L	470	215.3
Target	500	220
L, L2	470	207.5
H2, L2	535	207.5
H2, H	535	223.5

TABLE III: Corner Analysis for the waveguide simulation

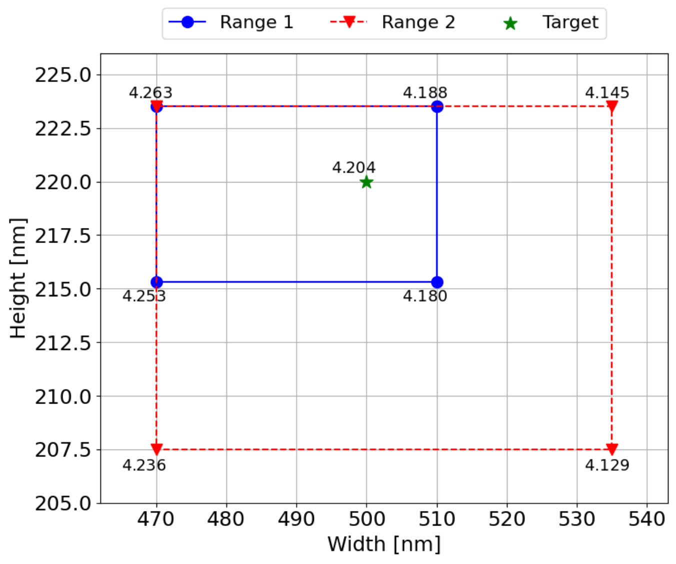

Fig. 9: Corner Analysis

The Lumerical MODE simulations for the Corner Analysis are shown in Table IV, where the calculated n_g uses Eq.2 and the Lumerical n_g is extracted directly from the simulation. Similarly, Fig.9 summarizes the range used for the corner analysis along with the simulated group index, where the minimum and maximum values are 4.129 and 4.263, respectively. A group index value was extracted from the experimental data of each MZI structure (refer to Table II) giving an average of 4.130±0.023. Hence, the experimental group index lies just within the corner analysis. This suggests that the waveguide may have an unusually thin silicon device layer (approximately 207.5 nm), and a width nearly 25 nm greater than the typical upper bound of 510 nm. These dimensions are atypical, which raises the possibility that the unexpectedly low group index could result from an overestimation of the MZI's path length difference, ΔL. If ΔL is longer than the nominal design value, the extracted group index would appear lower. Furthermore, Table V shows a trend where the group index decreases from approximately 4.15 to 4.12 as ΔL increases. This increase in ΔL also correlates with more waveguide bends, suggesting that waveguide bending may additionally contribute to the observed reduction in group index.

VII. Experimental Data

To characterize the devices, a custom-built automated test setup [2] [3] with automated control software written in Python was used [4]. An Agilent 81600B tunable laser was used as the input source and Agilent 81635A optical power sensors as the output detectors. The wavelength was swept from 1500 to 1600 nm in 10 pm steps. A polarization maintaining (PM) fibre was used to maintain the polarization state of the light, to couple the TE polarization into the grating couplers [5]. A 90^o rotation was used to inject light into the TM grating couplers [5]. A polarization maintaining fibre array was used to couple light in/out of the chip [6].

VIII. Analysis

A. Device Parameter Extraction from Experimental Data

The analysis of measurement data for each photonic component in the design was carried out using python scripts. The most important parts of these python scripts are included in the Appendices B and C. The first task was to import the measurement data from a csv file into a pandas dataframe for further data analysis, which was accomplished using the Python function `import_data_csv(folder_path)`.

1) Baseline and calibration Corrections: Once the data is imported, the `analysis_routine(...)` function was used to perform all data analysis steps in an automatic manner. The first step was to perform either a baseline or calibration correction to remove the "curved" amplitude response of the MZIs components and obtain a nearly flat MZI response, facilitating the subsequent data fitting. This curved response is due to the limited bandwidth of the grating couplers. In the case of the baseline correction, We perform a polynomial fitting on the MZI experimental data and subtract the fit from the raw data. Here, a polynomial of 4^{th} order was used and implemented using the `scipy` function `np.polyfit(x, y, 4)`. I chose to perform a polynomial fit, then trim the data based on a threshold amplitude [dB] value to ignore noisy data, and finally redo a polynomial fit on the trimmed data and subtract the fit from the raw data. The other method was to use a deembed (calibration) structure consisting of a pair of grating couplers connected by a strip waveguide. The experimental response of the deembed structure is subtracted from the response of each MZI component, which is referred as a calibration correction. Fig10 shows the response of the deembed structure **"ebramire_deembed1_1"** (shown in red) and the response for the MZI component **"ebramire_MZI3"** (shown in light blue), while the baseline and calibration corrections for the MZI response are shown in green and blue, respectively. The baseline correction shows a much flatter MZI response than the one obtained from a calibration correction. I decided to use this polyfit correction technique for the data fitting of all MZI structures.

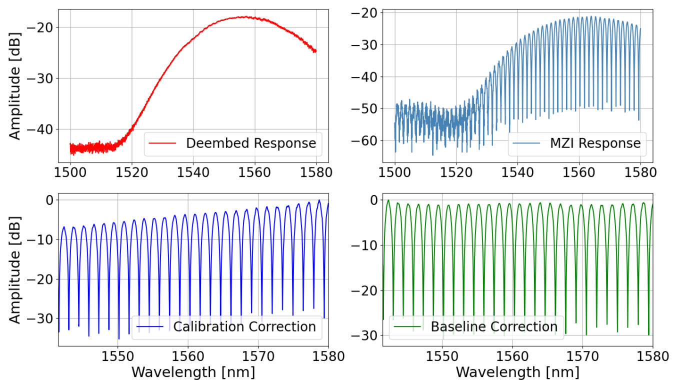

Fig. 10: Comparison between Baseline and Polyfit corrections

2) Estimating n_1: After the baseline correction, we perform a data fit on the experimental data using the theoretical transfer function for a MZI structure shown in Eq8. The data fit is performed using a nonlinear least of square fit where the fitting parameters are n_{eff}, α, and b, while ΔL is a constant with the corresponding value used in the design layout (refer to TableII). To improve the likelihood of convergence, the initial guess for n_{eff} is determined using Eq.1 and by estimating n_1 and n_2 from experimental data. The initial guess for n_1 is determined by calculating the integer mode number, N, for a destructive interference condition in Eq5 (i.e., $\frac{I_o}{I_i} = 0$). This destructive interference condition is evaluated at $\lambda = \lambda_o$ which reduces $n_{eff} = n_1$ based on Eq1 and therefore occurs when Eq9 is satisfied. The mode number is determined using the n_1 value from TableII and rounding to its nearest integer value for N. This mode number is plugged back into Eq9 to obtain a better estimate for n_1.

$$F = 10 \cdot \log_{10}\left(\frac{1}{4}\left|1 + \exp\left(-i\frac{2\pi n_{\text{eff}}}{\lambda}\Delta L - \frac{\alpha}{2}\Delta L\right)\right|^2\right) + b \quad (8)$$

$$\frac{2n_1\Delta_L}{\lambda_o} = 2N + 1 \quad (9)$$

3) Estimating n_2: The estimate for n_2 is obtained by using Eq2 at $\lambda = \lambda_o$, the previously determined n_1 initial guess, and an estimate for n_g. Rearranging Eq4 and adding the design values for Δ_L and $\lambda = \lambda_o$, one can solve for n_g using a value for the FSR ($\Delta\lambda$) that is extracted from experimental data. The latter is accomplished using my function called (`get_fsr_ng(...)`) which leverages the peak finding algorithm (`find_peaks(...)`) in the `scipy.signal` python library, akin to that available in MatLab. Once the wavelength values for all peaks are found in the experimental data, the average for the difference between neighbouring peaks is assigned as our FSR value and used to obtained our initial guess for n_2.

4) Data Fitting and Parameter Extraction: Since our photonic structures were design for the TE mode, we expect minimal change in dispersion as a function of wavelength, therefore we set n_3 equal to zero. Lastly, the loss term α is set to the value used during the circuit model simulations (5.9 dB/cm) and the amplitude offset parameter b is set to zero. Armed with initial guess values for all the fit parameters, a curve fit is performed using my function called (`data_fittng(...)`) which relies on the (`curve_fit(...)`) algorithm from the `scipy.optimize` library. Fig11 shows the baseline corrected data for the MZI3 component (black) overlaid with the data fit using the MZI transfer function model (red). Also, the experimentally extracted and calculated FSR and group index as a function of wavelength is included. There is a great agreement between the experimental data and the data fit. The same procedure was repeated for every MZI components using its appropriate ΔL and tweaking the initial parameters and/or the threshold amplitude cutoff in some instances to reach a similar level of agreement as that shown in Fig11. The extracted fitting parameters for each successful data fit are tabulated in TableV, including the average experimental FSR value and the range for simulated FSR value obtained from a corner analysis simulation. The average FSR fits within the corner analysis range in all cases except for MZI4. We see that on average the group index is 4.130 ± 0.023, which is in agreement with the corner analysis group index range.

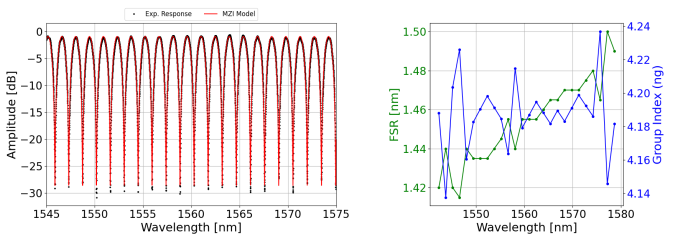

Fig. 11: Comparison between Baseline and Polyfit corrections

IX. Conclusion

In conclusion, the design and layout of several MZI structures was completed and then fabricated by Applied Nanotools Inc. The experimental data obtained from the fabricated chip was analyzed to extract the group index. It was demonstrated that the average extracted group index from several MZI was 4.130, which was in agreement with the range of group index obtained from a Corner Analysis. The waveguide dimensions used for the Corner Analysis seemed rather skewed from the nominal dimensions of the intended optical waveguide. It is suggested that perhaps the skewed dimensions used in the Corner Analysis as due to an overestimation of the path length difference and the influence of the number of waveguide bends.

Acknowledgment

I acknowledge the edX UBCx Phot1x Silicon Photonics Design, Fabrication and Data Analysis course, which is supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) Silicon Electronic-Photonic Integrated Circuits (SiEPIC) Program. The devices were fabricated by Richard Bojko at the University of Washington Washington Nanofabrication Facility, part of the National Science Foundation's National Nanotechnology Infrastructure Network (NNIN), and Cameron Horvath at Applied Nanotools, Inc. Omid Esmaeeli performed the measurements at The University of British Columbia. We acknowledge Lumerical Solutions, Inc., Mathworks, Mentor Graphics, Python, and KLayout for the design software.

References

[1] L. Chrostowski and M. Hochberg, *Silicon photonics design: from devices to systems*. Cambridge University Press, 2015.
[2] ——, "Chapter 12," in *Silicon Photonics Design: From Devices to Systems*. Cambridge, UK: Cambridge University Press, 2015, p. [Insert Page Range Here].
[3] Maple Leaf Photonics, "Maple leaf photonics," http://mapleleafphotonics.com, Seattle, WA, USA, 2024, accessed: 2024-04-11.
[4] M. Caverley, "Siepic probe station python code," http://siepic.ubc.ca/probestation, 2024, accessed: 2024-04-11.
[5] Y. Wang, X. Wang, J. Flueckiger, H. Yun, W. Shi, R. Bojko, N. A. Jaeger, and L. Chrostowski, "Focusing sub-wavelength grating couplers with low back reflections for rapid prototyping of silicon photonic circuits," *Optics express*, vol. 22, no. 17, pp. 20 652–20 662, 2014.
[6] PLC Connections, "Plc connections," http://www.plcconnections.com, Columbus, OH, USA, 2024, accessed: 2024-04-11.

Appendix A
Additional Plots and Tables

Width, Height	n_1	n_2 [1/μm]	n_3 [1/μm^2]	Calculated n_g	Lumerical n_g
H, H	2.4722	-1.107	-0.0466	4.18777	4.18796
H, L	2.4416	-1.121	-0.0333	4.17961	4.17979
L, H	2.4039	-1.199	-0.0529	4.26211	4.26250
L, L	2.3724	-1.213	-0.0341	4.25297	4.25337
Target	2.4468	-1.133	-0.0440	4.20359	4.20382
L, L2	2.3431	-1.221	-0.0110	4.23568	4.236064
H2, L2	2.4465	-1.085	-0.0128	4.12898	4.129044
H2, H	2.5072	-1.056	-0.0379	4.14461	4.144694

TABLE IV: Corner Analysis for the waveguide simulation

ID	ΔL [μm]	n_1	n_2 [1/μm]	n_3 [1/μm^2]	α [dB/cm]	b [dB]	Avg. n_g	Avg. FSR [nm]	Corner Analysis FSR Range [nm]
MZI1	100	2.4203	-1.147	-1.769x10^{-3}	11.8	-0.045	4.14641	5.794	5.634 - 5.808
MZI2	200	2.4382	-1.132	-4.101x10^{-7}	9.5	-0.781	4.15274	2.893	2.817 - 2.904
MZI3	400	2.4359	-1.131	-6.435x10^{-7}	4.0	-0.561	4.13183	1.454	1.41 - 1.456
MZI4	800	2.4350	-1.131	-3.357x10^{-2}	1.9	-0.211	4.12529	0.728	0.699 - 0.725
MZI5	1600	2.4378	-1.129	-4.135x10^{-2}	1.0	-0.007	4.12055	0.364	0.353 - 0.365
MZI6	3200	2.4385	-1.129	-6.032x10^{-8}	0.7	0.213	4.11640	0.182	0.176 - 0.182
MZI7	14150	2.4379	-1.128	-4.156x10^{-8}	3.4	-1.897	4.11778	0.041	0.040 - 0.042

TABLE V: Extracted parameters from MZI devices

Appendix B
Python Script: Helping Functions

```python

import pandas as pd
from io import StringIO
import matplotlib.pyplot as plt
import numpy as np
import os
from scipy.optimize import curve_fit
from scipy.signal import find_peaks


# Imports data from csv files and outputs a dataframe structure for further data analysis
def import_data_csv(folder_path):

    # List all CSV files in the folder
    csv_filename = [f for f in os.listdir(folder_path) if f.endswith(".csv")]
    csv_file_path = os.path.join(folder_path, csv_filename[0])
    print(f'>>csv file path: {csv_file_path}')

    # Here I expect to have a single csv file per folder. If more csv folders are found then an error is
        thrown
    if len(csv_filename) == 1:
        # Load file
        with open(csv_file_path, "r", encoding="utf-8") as f:
            lines = f.readlines()

        # Filter out metadata lines (those starting with "#")
        data_lines = [line for line in lines if not line.startswith("#") and "," in line]

        # Reconstruct the CSV content using only data lines
        csv_content = "\n".join(data_lines)

        # Read as a CSV using the first line as header
        df_raw = pd.read_csv(StringIO(csv_content))

        # Transpose the DataFrame
        df = df_raw.set_index('wavelength').T.reset_index()

        # Rename columns
        df.columns = ['Wavelength', 'Channel 1', 'Channel 2', 'Channel 3', 'Channel 4']

        # Convert all values to numeric
        df = df.astype(float)

        print(f'>> Headers: {df.columns}')

        return df
    else:
        print(f'too many csv files inside folder {folder_path}')


# Model for MZI transfer function based on lectures from eDX course
def mzi_func(lambda_x, n1, n2, n3, alpha, b, lambda_o, delta_L):
    n_eff = n1 + n2 * (lambda_x - lambda_o) + n3 * (lambda_x - lambda_o) ** 2
    trans_func = 10 * np.log10(1/4 * np.abs(1 + np.exp(-1j * (2 * np.pi * n_eff / lambda_x) * delta_L) -
        alpha * delta_L / 2) ** 2) + b
    return trans_func


# Performs a nonlinear least of squares fit of the experimental data using an initial guess for the fit
    parameters and
# fitting bounds for each parameter.
def data_fitting(df, x_label, y_label, fit_const, initial_guess, bounds, maxfev, plot_opt=False):

    # Experimental Data. X data is turned into meters
    x_data = df[x_label] * 1e-9
    y_data = df[y_label]

    # Constant values passed into fitting functions
    lambda_o, delta_L = fit_const

    # Fitting function: uses lambda to handle fitting parameters and constants
    fit_func = lambda x, n1, n2, n3, alpha, b: mzi_func(x, n1, n2, n3, alpha, b, lambda_o, delta_L)

    # Performs fit using curve_fit method
    popt, pcov = curve_fit(fit_func, x_data, y_data, p0=initial_guess, bounds=bounds, maxfev=maxfev)
    print(f"Initial Guesses - n1: {initial_guess[0]}, n2: {initial_guess[1]}, n3: {initial_guess[2]}, alpha
        : {initial_guess[3]/1e2}[dB/cm], b: {initial_guess[4]}")

    if plot_opt:
        # Plots the fit results
        plt.scatter(x_data, y_data, label='Data', color='blue', s=0.5)
        plt.plot(x_data, mzi_func(x_data, *popt, *fit_const), color='red', label='Fit')
        plt.legend()
        plt.show()

    return popt, pcov



# Extracts FSR & group index from the experimental data by finding peaks and calculating their difference
    in wavelength
def get_fsr_ng(df, x_label, y_label, delta_L, height=5, distance=20, prominence=5, plot_opt=False):

    # Data used to find peaks. The Y data is reverse in magnitude
    y_data_inv = -df[y_label].to_numpy()
    x_data = df[x_label].to_numpy()

    # Finds peaks in the y data using the find_peaks function from scipy. Uses height, distance and
        prominence
    # parameters to tune the peak search
    peaks, properties = find_peaks(y_data_inv, height=height, distance=distance, prominence=prominence)

    # Loops over each pair of neighbouring peaks and calculates their difference
    fsr_lambda, fsr, ng = [], [], []
    for lambda_i1, lambda_i2 in zip(x_data[peaks][:-1], x_data[peaks][1:]):

        # Calculates average wavelength between pair of peaks which is needed for group index calculation
        lambda_avg = np.average([lambda_i2, lambda_i1]) * 1e-9
        fsr_lambda.append(lambda_avg)
        # Calculates difference wavelength between pair of peaks and assigns it as the fsr value
        lambda_delta = np.abs(lambda_i2 - lambda_i1) * 1e-9
        fsr.append(lambda_delta)
        # Calculates group index
        ng.append(lambda_avg ** 2 / (lambda_delta * delta_L))

    # Creates a dataframe with the fsr and group index data vs. wavelength
    df_fsr = pd.DataFrame({'Wavelength': fsr_lambda, 'FSR': fsr, 'ng': ng})

    # Plots relevant data to extract the fsr and group index data
    if plot_opt:

        # Plots the fsr and group index data vs. wavelength
        plot_inspection([df_fsr, df_fsr], ['Wavelength']*2, ['FSR', 'ng'], y_lims=[[]]*2, x_lims=[[]]*2,
                        rows=1, cols=2, title='FSR & ng')

        # Plots the results of peak searching
        plt.plot(x_data, y_data_inv, label='Raw Signal')
        plt.plot(x_data[peaks], y_data_inv[peaks], 'rx', label='Peaks')
        plt.legend()
        plt.title("Detected Peaks")
        plt.show()

    return df_fsr
```

Listing 1: Example Python Script for Data Fitting

Appendix C
Python Script: Data Analysis

```
import numpy as np
import os
import help_functions as hlp_fun


# Analysis data from an optical component by performing the following:
# (1) Baseline correction (using polynomial fit or deembeding fit)
# (2) Numerically determines optimal initial guesses for n1 and n2 parameters from the n_eff(lambda)
    expression
# (3) Performs a curve fit of "baseline-corrected" experimental data with the MZI transfer function model
    using initial guesses for n1, n2, n3, alpha, and b (offset param)
# (4) Returns fit results for the waveguide and MZI parameters (n1, n2, n3, ng, fsr)
def analysis_routine(working_path, folder_deembed, folder_component, y_deembed_label, y_label,
    corr_opt_index, delta_L, pks_height, pks_distance, pks_prominence, amplitude_cutoff,
    remove_fsr_outliers, plot_opt, overwrite_initial_guess, lower_percent=1, upper_percent=1, maxfev=10000,
     y_scale_linear=False):

   ...

    # Performs polynomial fit on optical component data
    if correction_chosen == 'poly_fit':

        # Inspect data to choose correct channel
        if plot_opt:
            hlp_fun.plot_all_channels(df, y_scale_linear)

        # Performs a polynomial fit on data
        x, y_poly_fit = hlp_fun.fit_poly_2_data(df, x_label='Wavelength', y_label=y_label, y_bound=0,
                                                filter_y_opt=False, plot_opt=plot_opt)
        df['Poly_fit'] = y_poly_fit

        # Trims y data to ignore noisy data
        df_filter = hlp_fun.trim_y(df, 'Poly_fit', y_bound)

        # Performs polynomial correction
        df_flatten, header_y_flatten = hlp_fun.subtract_poly(df_filter, x_label='Wavelength', y_label=
            y_label, y_bound=y_bound, filter_y_opt=False, plot_opt=plot_opt)

        # Inspects correction results
        if plot_opt:
            hlp_fun.plot_channel(df_flatten, x_label='Wavelength', y_label=header_y_flatten)
            hlp_fun.plot_channel(df_flatten, x_label='Wavelength', y_label=y_label)

        df_2 = df_flatten
        y_label2 = header_y_flatten
```

Listing 2: Example Python Script for Data Fitting