FACULTY OF ENGINEERING AND ARCHITECTURE

Programmable Photonic Circuits: From MEMS Building Blocks to Software Control

Iman Zand

Doctoral dissertation submitted to obtain the academic degree of
Doctor of Photonics Engineering

Supervisors

Prof. Wim Bogaerts, PhD* - Muhammad Umar Khan, PhD**

* Department of Information Technology
Faculty of Engineering and Architecture, Ghent University

** Keysight Technologies

December 2024

GHENT UNIVERSITY

ISBN 978-94-6355-939-3
NUR 958, 959
Wettelijk depot: D/2024/10.500/144

Members of the Examination Board

Chair

Prof. Luc Dupré, PhD, Ghent University

Other members entitled to vote

Cleitus Antony, PhD, Tyndall National Institute, Ireland
Prof. Em. Roel Baets, PhD, Ghent University
Marcus Dahlem, PhD, imec
Prof. Dirk Stroobandt, PhD, Ghent University
Prof. Kasper Van Gasse, PhD, Ghent University

Supervisors

Prof. Wim Bogaerts, PhD, Ghent University
Muhammad Umar Khan, PhD, Keysight Technologies

Acknowledgements

First and foremost, I would like to express my deepest gratitude to my advisor, Wim Bogaerts, for his exceptional guidance, encouragement, and unwavering support throughout my Ph.D. journey. His expertise and insights have been pivotal in shaping this research and helping me grow as a researcher. Over the course of my doctoral work, we navigated both technical and non-technical challenges, and I gained invaluable lessons from his problem-solving approach and our interactions.

I am profoundly thankful to the members of my jury, Prof. Em. Roel Baets, Prof. Dirk Stroobandt, Prof. Kasper Van Gasse, Dr. Marcus Dahlem, and Dr. Cleitus Antony, for their time, effort, and invaluable feedback on my thesis. Their thoughtful suggestions, critical insights, and rigorous evaluations have significantly improved the quality of this work and inspired me to view my research from new perspectives.

Being part of the MORPHIC project was an incredible experience, and I am deeply grateful to my MORPHIC consortium partners. Collaborating with such a diverse and knowledgeable team was a privilege. I would especially like to thank Pierre, Yuji, Hamed, Banafsheh, Auron, and Dr. Antony for their contributions and support.

I would also like to thank my colleagues in the Photonic Research Group (PRG). A special thanks to Umar for his technical discussions, lab training, and admirable calmness, which I greatly respect. I will fondly remember Xiangfeng and our endless discussions about both science and life. My gratitude also goes to Hong, Lukas, Mi, Antonio, Yu, Khannan, Amin, Ivo, and Jing for being such wonderful colleagues. I sincerely thank Clemens and Jasper for their lab management and technical assistance, as well as Steve and Muhammad for their cleanroom training and support. Additionally, I am deeply appreciative of Kristien, Mike, Ilse Van Royen, and Ilse Meersman for their kindness and unwavering support.

I am also thankful to other friends at UGent and in the U.S. who may not be mentioned by name but have offered their friendship and support throughout this journey. Lastly, and most importantly, I owe immense gratitude to my family for their unconditional love, encouragement, and belief in me. Your support has been

the foundation of my success—thank you for always being there for me!

Ghent, December 2024
Iman Zand

Contents

List of Figures

List of Tables

List of Acronyms

A

ADC	Analog-to-Digital Converter
API	Application Programmer Interface
ASPIC	Application Specific Photonic Integrated Circuit

B

BB	Beaglebone
BBB	Beaglebone Black
BEOL	Back End of the Line
BHF	Buffered Hydrofluoric acid
BOX	Buried oxide
BPD	Balanced Photodetector or photodiode
BUL	Basic Unit Length
BW	Bandwidth
b-HF	Buffered Hydrogen Fluoride

C

CLI	Command Line Interface
CMOS	Complementary metal–oxide–semiconductor
CP	Coupler

D

DAC	Digital-to-Analog Converter
DC	Direct Current
DC	Directional Coupler

E

EIC	Electronic Interface and Control/Electronic Integrated Circuits
EO	Electro-Optic
ER	Extinction Ratio

F

FA	Fiber Array
FEOL	Front End of the Line
FIR	Finite impulse response (filter)
FPGA	Field Programmable Gate Array
FPPA	Field Programmable Photonics Array
FP-PIC	Field Programmable Photonics Integrated Circuit
FSR	Free Spectral Range

G

GC	Grating Coupler
GUI	Graphical User Interface

H

HF	Hydrofluoric acid

I

IC	Integrated Circuit
iSiPP50G	Imec's Silicon Photonics Platform
IIR	Infinite Impulse Response
IL	Insertion Loss
InP	Indium Phosphide
IO	Input/output
IP	Internet Protocol
IR	Infrared
ITO	Indium Tin Oxide

L

LC	Liquid Crystal
LiDAR	Light Detection and Ranging
LSI	Large-Scale Integration

M

MEMS	Micro-Electro-Mechanical-Systems
MIR	Mid-Infrared
MMI	Multi-mode Interferometer
MORPHIC	MEMS-based zerO-poweR PHotonic Integrated Circuits
MPW	Multi-project Wafer
MRR	Micro Ring Resonator
MSI	Medium-Scale Integration
MWP	Microwave Photonics
MZI	Mach Zehnder Interferometer

O

OFDR	Optical Frequency-Domain Reflectometry
OVA	Optical Vector Analyzer

P

PC	Personal Computer
PCB	Printed Circuit Board
PCell	Parametric Cell
PD	Photodetector or Photodiode
PDK	Process Design Kit
PIC	Photonics Integrated Circuit
PI-loss	Path-independent loss
PS	Phase Shifter

R

RF	Radio-Frequency
RR	Ring Resonator

S

SBC	Single Board Computer
SEM	Scanning Electron Microscope
SiN	Silicon Nitride
SiPh	Silicon Photonics
SOA	Semiconductor Optical Amplifier
SOI	Silicon on Insulator
SPI	Serial Peripheral Interface
SSI	Small-Scale Integration

T

TCP	Transfer Control Protocol
TIA	Trans-impedance Amplifier

V

VHDL	VHSIC Hardware Description Language
vHF	Vapour-phase Hydrofluoric acid
VLSI	Very Large-Scale Integration

W

WDM	Wavelength division Multiplexing
WG	Waveguide
WSS	Wavelength selective Switch

Samenvatting

1 Siliciumfotonica

De integratiedichtheid en het industriële momentum hebben van siliciumfotonica een ideaal platform gemaakt om fotonische geïntegreerde circuits (Photonic Integrated Circuit, PIC) in te bouwen. Silicium PIC's vinden langzaamaan hun plaats in biomedische- en omgevingssensoren, LiDAR-bouwstenen, computationele versnellers en kwantuminformatiesystemen. Door het hoge brekingsindexcontrast en de compatibiliteit met geavanceerde CMOS-productieprocessen te benutten, wordt er consequente vooruitgang geboekt op vlak van zowel kleinschalige integratie (Small-Scale Integration, SSI) als zeer grootschalige integratie (Very-Large-Scale-Integration, VLSI), met demonstraties van circuits die uit honderden componenten bestaan [1]. Hoewel sommige functies, zoals laag vermogen elektro-optische faseverschuivers, geïntegreerde lichtbronnen[1] en optische isolatoren, nog niet matuur zijn op siliciumfotonica-platforms, worden er enorme inspanningen geleverd om de zwakheden van siliciumfotonica weg te werken.

2 Programmeerbare Fotonische Geïntegreerde Circuits

Een recent ontwikkelde categorie binnen fotonische geïntegreerde circuits is deze van programmeerbare fotonica [11–20]. Programmeerbare PIC's zijn fotonische circuits gemaakt uit golfgeleiders die afstembare componenten zoals koppelaars en faseshifters verbinden. De hoofdtaak van de koppelaars is om het vermogen in een voortbewegende lichtbundel op te splitsen in twee verschillende paden. De faseverschuivers moeten de golffase van het voortbewegende licht aanpassesn, door het licht een beetje te versnellen of te vertragen. Door deze afstembare optische componenten op de fotonische chip te koppelen aan elektronische hardware, kunnen we elektrische signalen van elektronische aanstuurcircuits naar de chip overbrengen via commando's die gegenereerd worden vanuit een softwarebibliotheek. Hierdoor

[1] Sommige siliciumplatformen bieden reeds integratie van bepaalde lichtbronnen aan - Openlight, Tower, INTEL.

kunnen we lichtinterferenties op de chip aansturen en diverse wiskundige taken uitvoeren zoals lineaire matrixbewerkingen [11, 12, 21] of functies implementeren zoals golflengtefilters [17].

De motivatie achter de ontwikkeling van programmeerbare fotonica komt voort uit verschillende eigenschappen van deze circuits. De programmeerbaarheid van dergelijke circuits kan de "yield" (i.e. de fractie succesvol gefabriceerde chips) verhogen en maakt de implementatie van complexere functies mogelijk. Het is namelijk zo dat, door het nauwkeurig afstemmen van elke component op de chip, fouten die ontstaan door fabricatievariaties gecorrigeerd kunnen worden, waardoor de waarschijnlijkheid dat de chip correct functioneert wordt verhoogd. Daarnaast kunnen met programmeerbare ontwerpen, zoals schakelmatrices, fotonische versnellers voor matrixbewerkingen of kwantum-optische circuits worden gecreëerd voor meerdere toepassingen, in plaats van beperkt te zijn tot slechts één toepassing, zoals tegenwoordig typisch het geval is.

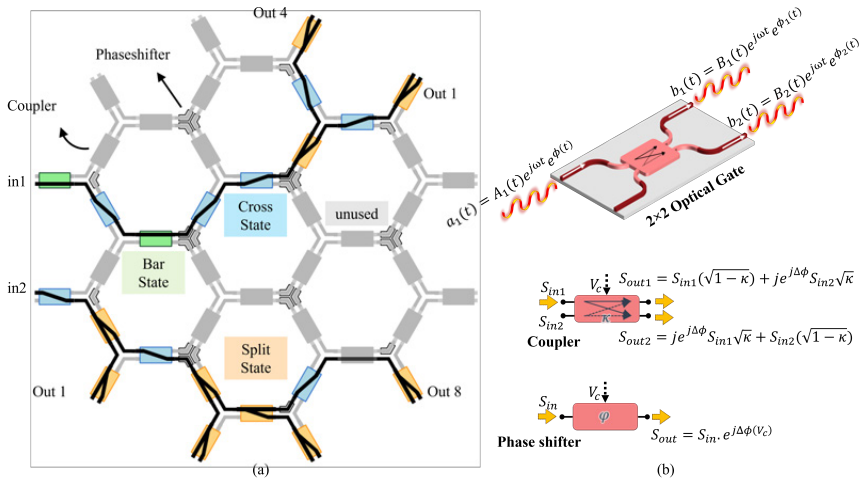
Een andere belangrijke drijfveer voor programmeerbare fotonica is de opkomst van algemene "Field Programmable Photonic Integrated Circuits"[2], ofwel fotonische processoren die voor meerdere doelen kunnen worden ingezet (FP-PIC's). Deze chips zijn ontworpen om veelzijdige chips te kunnen worden geprogrammeerd voor verschillende functies [22]. FP-PIC's vertegenwoordigen een significante vooruitgang in de verspreiding van fotonische circuits, aangezien ze prototypingprocessen mogelijk maken die vergelijkbaar zijn met die van elektronische FPGA's. Deze circuits gaan verder dan alleen fotonische chips, waarbij de integratie van elektronica, verpakkingstechnologieën, en een softwareprogrammeringsinterface (net als andere programmeerbare circuits) vereist is.

3 Programmeerbare Fotonische Geïntegreerde Circuits voor Algemene Doeleinden

Programmeerbare circuits worden ingedeeld in twee groepen op basis van hun verbindingsarchitecturen, of "mesh". Dit is het netwerk van golfgeleiders dat de afstembare bouwblokken van het circuit verbindt. Deze twee types mesh zijn "voorwaartspropagerendën "circulerend". In voorwaartspropagerende meshes beweegt het lichtsignaal van de ingangsgolfgeleider naar de uitgangsgolfgeleider zonder lussen of terugkoppeling. In circulerende meshes kan het lichtsignaal binnen het circuit in elke richting circuleren met behulp van golfgeleiderlussen.

FP-PIC's zijn gebouwd op basis van circulerende meshes, die kunnen variëren in verbindingstopologie, zoals rechthoekig of hexagonaal, en in vorm, zoals rechthoekig, radiaal of parallelogram. Zo is het mesh dat wordt getoond in Fig. 1a een

[2]Multipurpose programmeerbare fotonische processors, of veldprogrammeerbare fotonische gatearrays.



Figuur 1: a) schema van een 7-cel hexagonaal mesh, waar twee verschillende functies worden uitgevoerd: een 1×6 tap-gebaseerde bundelsturing ("beamformer"), en een 1×4 boom-gebaseerde bundelsturing, b) een 2×2 optische poort, en construerende koppelaar en faseverschuiver.

7-cel hexagonaal mesh met een radiale configuratie.

Een ander belangrijk concept in programmeerbare PIC's zijn de 2×2 optische poorten (Fig. 1b) die nodig zijn om de connectiviteit en interferenties in het golfgeleidermesh te configureren. Ze kunnen worden geïmplementeerd als afstembare directionele koppelaars [2, 23] of als Mach-Zehnder interferometers met faseverschuivers [24]. Optische poorten stellen ons in staat om licht van twee ingangsgolfgeleiders te mengen in twee uitgangsgolfgeleiders en de fasevertraging tussen hen te regelen. Naast optische poorten zijn monitordectectoren essentiële elementen binnen het programmeerbare circuitmesh en deze moeten strategisch doorheen het ganse mesh worden verdeeld. Ze zijn cruciaal voor het handhaven van de stabiliteit van het circuit in de geprogrammeerde toestand. Deze monitoren observeren optisch vermogen (en fase) binnen het circuitmesh zonder significante optische verliezen te introduceren [25, 26].

4 Uitdagingen bij het Opschalen van Programmeerbare Fotonica

Een ideaal FP-PIC zou moeten kunnen opschalen tot een zeer grote omvang, en een groot aantal optische poorten bezitten, wat uitgebreide functionaliteit mogelijk maakt. Echter, in de realiteit worden we geconfronteerd met praktische

beperkingen voor de grootte en de prestaties van onze circuits. Een van de kernvragen in het veld van FP-PIC's is hoe we de hoeveelheid schakelelementen kunnen opschalen en welke haalbare oplossingen bestaan er om beperkende factoren te overwinnen. Optisch verlies, het energieverbruik van de circuitcomponenten en verpakkingsbeperkingen behoren tot de belangrijkste parameters bij de realisatie van programmeerbare PIC's, vooral wanneer we geïnteresseerd zijn om de circuits op te schalen.

Voor silicium-PIC's hebben onderzoekers succesvol de golfgeleiderpropagatieverlies teruggebracht van $20.0\,\mathrm{dB/cm}$ in eerste demonstraties tot $0.5\,\mathrm{dB/cm}$ in recentere ontwikkelingen [27, 28]. Deze vooruitgang weerspiegelt duidelijk de verbeteringen in de fabricageprocessen. Daarnaast kunnen nieuwe topologieën en ontwerpstrategieën worden gebruikt om geoptimaliseerde geometrieën te implementeren, die andere procesgerelateerde verbeteringen aanvullen. Bijvoorbeeld, een rechte golfgeleider kan soepel overgaan in een cirkelvormige bocht met behulp van een Spline-transitiebenadering of een Euler-bocht, die mode-mismatch minimaliseert zonder significant de voetafdruk te vergroten of een extra verlies te introduceren [29]. Naast golfgeleiders moeten ook roosterkoppelaars (die dienen als optische poorten voor het verzenden en ontvangen van lichtsignalen naar en van de fotonische chip) en glasvezelconnecties geoptimaliseerd worden om de optische verliezen van de circuits te minimaliseren.

Een van de belangrijkste uitdagingen in FP-PIC's is de ontwikkeling van een breedbandig, laag-verlies en compact actuatiesysteem. Momenteel maakt de meest voorkomende methode gebruik van het thermo-optische effect, door lokaal actuatoren op te warmen. Op verwarming gebaseerde faseshifters kunnen geïmplementeerd worden over een relatief korte afstand ($<50\,\mu\mathrm{m}$) en zijn compatibel met monolithische integratie in siliciumfotonische fabricageplatformen, waarbij lage optische verliezen worden gehandhaafd. Hun energieverbruik is echter minimaal $1\,\mathrm{mW}$ per actuator (vaak $5\,\mathrm{mW}$-$20\,\mathrm{mW}$).

MEMS-gebaseerde actuatie biedt een alternatieve methode om buitensporig energieverbruik en hoge optische verliezen in individuele bouwblokken te verminderen, waardoor de dichtheid van schakelingen in grootschalige fotonische geïntegreerde circuits kan worden opgeschaald. Deze aanpak vereist het aantonen van MEMS-compatibiliteit met gevestigde fabricageplatforms en het integreren ervan met hoogwaardige standaardcomponenten. Het is weliswaar essentieel dat de prestaties van deze MEMS-actuatoren gelijkwaardig of superieur zijn vergeleken met andere types actuatoren.

5 Het MORPHIC-Project

Het MORPHIC-project[3] (2018-2021) was een initiatief van Europese Horizon 2020-programma, gericht op het integreren van MEMS-actuatoren in siliciumfotonica om multifunctionale programmeerbare PIC's (FP-PIC's) te ontwikkelen. Het project richtte zich op belangrijke beperkingen in siliciumfotonica, zoals variabiliteit en hoog energieverbruik, door gebruik te maken van MEMS actuatoren met een laag vermogen, of zelfs een niet-vluchtige schakeltoestand, dit met het oog op betere prestaties dan traditionele, op verwarming gebaseerde, afstemactuatoren. Aanzienlijke vooruitgang werd geboekt bij het integreren van MEMS in het IMEC iSiPP50G-platform zonder de bestaande functionaliteiten te beïnvloeden, evenals bij het ontwikkelen van compacte MEMS-gebaseerde actuatoren. Hierbij werd in het project ook een techniek ontwikkeld om volledige siliciumplakken ("wafers") hermetisch af te dichten, en werden er ook transitieschakelingen ("interposers") ontworpen en gefabriceerd om circuits met hoge dichtheid te verbinden met de buitenwereld.

In dit project was onze onderzoeksgroep verantwoordelijk voor de projectcoördinatie, het ontwerpen en testen van kleine en grote programmeerbare circuits, het karakteriseren van circuitcomponenten zoals golfgeleiders en overgangen, het plannen van de lay-out van de MORPHIC-chips, het samenwerken met partners aan systeemcomponentherontwerp en -aanpassingen, en het ontwikkelen van een technologie-ecosysteem in het algemeen. Mijn persoonlijke bijdragen omvatten het ontwerpen en karakteriseren van FP-PIC-circuits, het ontwikkelen van een softwareraamwerk voor FP-PIC-ontwerp, -analyse en -aansturing, samenwerkingen met Tyndall om een netlist-database te creëren voor elektronische en optische componenten, het testen van elektronische geïntegreerde schakelings(EIC)-borden, het ontwerpen van kleinschalige interposers en PCB-interconnects, het geven van feedback aan MEMS-ontwerpers bij EPFL en KTH voor componentoptimalisatie, het assisteren bij het schrijven van rapporten en projectleveringen, en het deels coördineren van werkpakketvergaderingen gerelateerd aan mijn onderzoek.

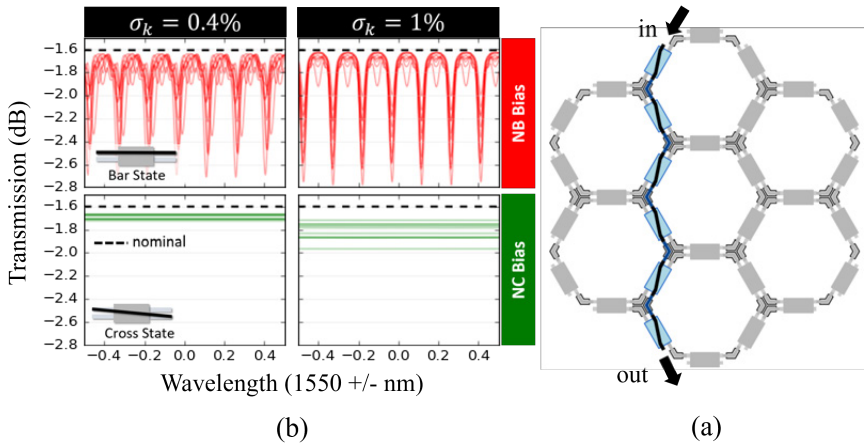
6 Mesh-analyse

Voor het ontwerpen van de optische kern van de FP-PIC-circuits hebben we verschillende meshanalyses uitgevoerd, waaronder verlies- en schalingsevaluaties op basis van de kritieke parameters en opgemeten prestatiegegevens van MORPHIC MEMS-bouwblokken. We hebben ook verschillende meshvormen bestudeerd, zoals rechthoekig en radiaal. Uit onze analyse blijkt dat rechthoekig gevormde meshes met een hexagonale topologie de kleinste voetafdruk innemen (voor hetzelfde aantal meshcellen) om verschillende Toepassingsgerichte functies zoals schake-

[3] https://h2020morphic.eu/

laarmatrices en bundelvormers (“beamformer”) te ondersteunen. Terwijl radiaal gevormde meshes betere verliesmetrieken vertonen dan rechthoekige, nemen ze een aanzienlijk grotere voetafdruk in om dezelfde functionele circuits te accommoderen, waardoor hun voordeel in verliesprestaties teniet wordt gedaan.

Een van de studies die we uitgevoerd hebben betreft het verminderen van de parasitaire effecten in een golfgeleidernetwerk veroorzaakt door afwijkend gedrag van de koppelaars en faseverschuivers. In het bijzonder, in circuits met terugkoppeling (lussen) kunnen deze parasiteire effetcen secundaire en tertiarie verbindingen tot stand brengen, die aanleiding kunnen geven tot ongewenste interferenties en resonanties in de frequentierespons van het circuit [30]. Met behulp van Monte-Carlosimulaties hebben we aangetoond dat een correct instelling van de ongebruikt koppelaars deze parasitaire effecten drastisch kunnen inperken of zelfs elimineren. Als voorbeeld toont Fig. 2 de transmissie van een recht vebindingspad in een 7-cellsnetwerk waarbij de ongebruikte koppelaars voorgeprogrammeerd zijn in de bar-, dan wel in cross-toestand.

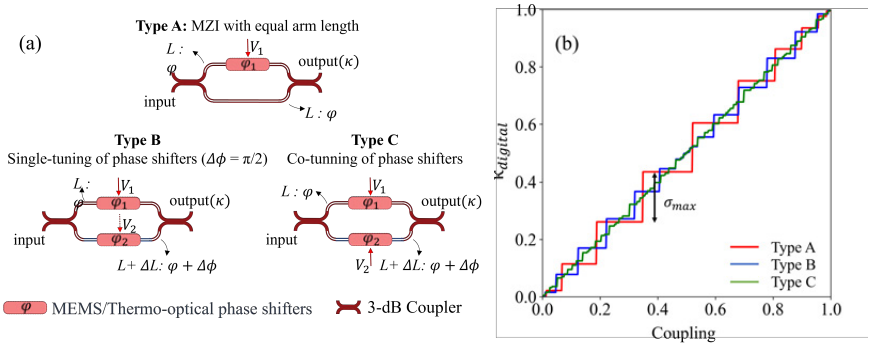


Figuur 2: (a) Schematisch verbindingsdiagram van een gerouteerd pad (met $L_{path} = 6.L_u$) in een 7-cel-netwerk. (b) Transmissiespectra van het netwerk voor twee configuraties van voorgeprogrammering van de ongebruikte koppelaars, repectievelijk in 'bar' (NB, rood) en 'cross' (NC, groen). De resultaten zijn weergegeven voor $\sigma_\kappa = 0.4\%, 1.0\%$ van links naar rechts.

We hebben een modelschema ontwikkeld om de fouten in programmeerbare fotonische circuits te beoordelen vanwege onvolkomenheden in de faseverschuivers en afstembare koppelaars. Deze kunnen veel oorsprongen hebben, maar één die we in het bijzonder overwogen, was de discretisatiefout die wordt veroorzaakt door de digitaal-naar-analoog omvormer (DAC). We hebben verschillende architecturen bestudeerd voor zowel MEMS-gebaseerde als op verwarming gebaseerde 2×2 afstembare MZI-gebaseerde koppelaars om de digitaliseringsfouten veroorzaakt door DAC's te verkleinen. Type A is een MZI met gelijke armlengtes en een aktuator

in een van de armen. In Type B, een van de armen heeft een $\pi/2$ faseverschuiving (in kwadratuur) met aktuatoren in beide armen. In type C worden deze beide aktuatoren samen gebruikt on zo een discrete 2-D ruimte defini eren voor de koppeling.

Figuur 3b demonstreert de resons van $\kappa_{digital}$ in de drie bovengenoemde configuraties, vergeleken met de ideale respons, en dit voor een spanningsaansturing met een precisie van 4-bit. Vergelijking van de curves toont duidelijk aan dat de MZIin kwadratuur waarbij beide armen gelijktijdig worden aangestuurd (type C) de accuraatheid van de koppeling drastisch verbetert. Dit kan afgeidl worden uit de maximale stapgrootte van de koppelijk (σ_{max}, aangeduid met de zwarte pijl) voor deze situaties.



Figuur 3: Afstembare 2 × 2 MZI-koppelaar met faseverschuivers. Type A: MZI met gelijke armlengte met een enkele aktuator in een van de armen. Type B: MZI met $\pi/2$ fasevertraging (kwadratuur) met aktuatoren in beide armen, waarbij de koppelaar aangedreven wordt in 'push-pull', door telkens één van de faseverschuivers aan te sturen. Type C: MZI zoals type B, maar waarbei beide faseverschuivers tegelijk worden aangestuurd, waarbij de aansturingsruimte tweedimensionaal wordt.

Door het aantal bouwblokken (koppelaars, faseverschuivers, I/O, en laagfrequente elektrische verbindingen) per opstelling te vergelijken, te analyseren hoe hun aantallen op te schalen, en om eveneens te onderzoeken hoe het verlies van het kortste pad daarbij meeschaalt, hebben we een uitgebreide kaart gecreëerd voor het implementeren van de programmeerbare meshes.

7 Schakelingsontwerp

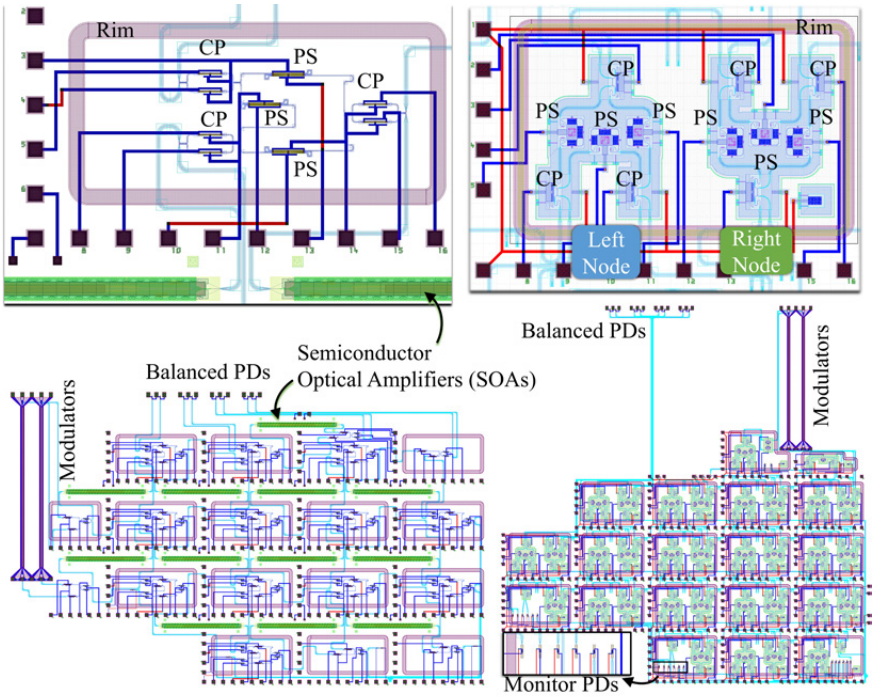
We hebben twee kleinschalige FP-PIC-demonstratoren geïmplementeerd voor het MORPHIC-project:

- Een 7-cel hexagonaal mesh geactiveerd door verwarmingselementen. We hebben ook de gelegenheid aangegrepen om halfgeleider-optische-versterkers

(SOA) in dit mesh te integreren, om te onderzoeken hoe deze kunnen helpen de verliezen in grote programmeerbare meshes te compenseren.

- Een 24-cel hexagonaal mesh met MEMS-koppelaars en -faseverschuivers.

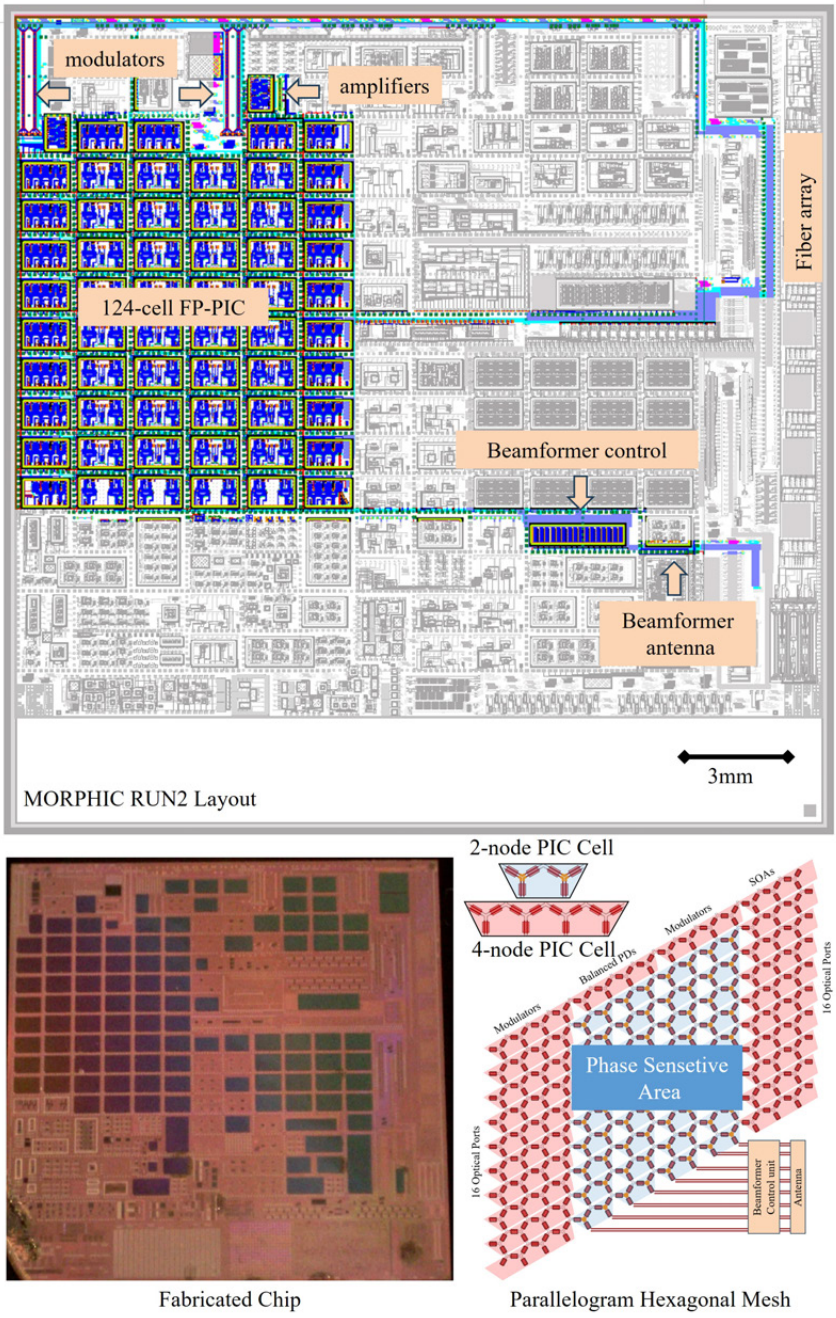
Beide schakelingen zijn geïllustreerd in Fig. 4 Ze zijn ook verbonden met twee modulatoren, 4 paren van gebalanceerde fotodiodes, en 8 vezelpoorten.



Figuur 4: Kleinschalige FP-PIC ontwerpen op MORPHIC RUN2. Links: 7-cel hexagonaal mesh gebaseerd op warmteëlementen, met optionele optische versterkers. Rechts: 24-cel MEMS-gebaseerd mesh.

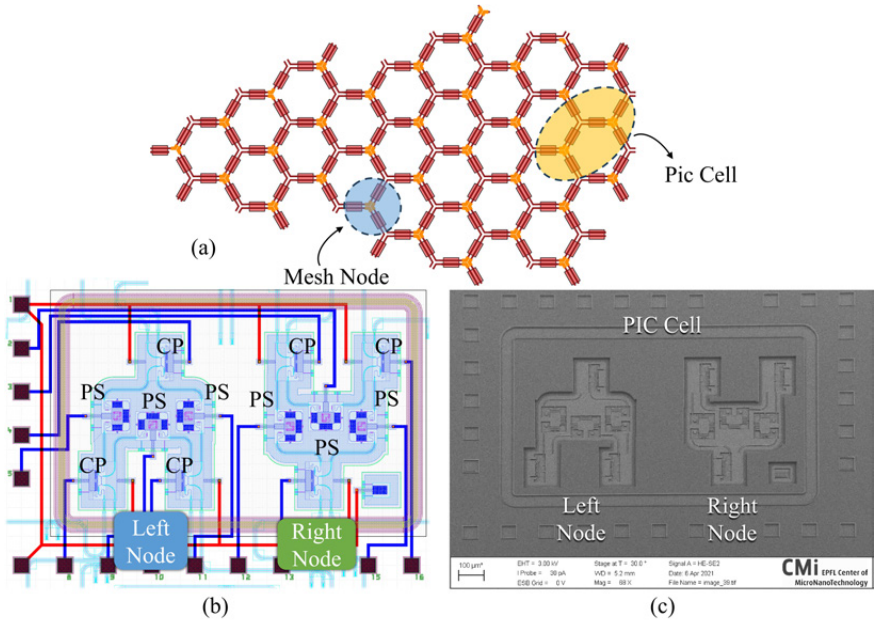
Deze ontwerpen werden gevolgd door een volledig demonstratorcircuit in de volgende fabricagerun. Voor ons grootschalige FP-PIC-ontwerp ($oppervlakte > 1\,\mathrm{cm}^2$, met ongeveer 1000 elektrische verbindingen), hebben we een 126-cel FP-PIC geïmplementeerd met behulp van een 14×9 parallelogramvormig mesh (Fig. 5). De verandering van een rechthoekige naar een parallelogramvorm werd gemaakt om een schonere plattegrond op de chip te bereiken, aangezien deze kon worden gerangschikt als een set van rechthoekige cellen. De vezelingangen (16) en -uitgangen (16) bevinden zich repectievelijk aan de west- en oostzijde. De noordzijde is verbonden met 4 modulatoren en 4 gebalanceerde fotodetectoren, en we hebben ook twee ruimtes voorzien voor de integratie van halfgeleider-optische-versterkers (SOA) die

kunnen aangebracht worden met behulp van de transfer-print methode [8]. Aan de zuidzijde hebben we extra monitorfotodiodes opgenomen en 16 uitgangen zijn verbonden met een pad-gebalanceerde gefaseerde array-antenne.



Figuur 5: Grootschalig FP-PIC met een 126-cel mesh: Indeling, schema, en de gefabriceerde chip.

Om de FP-PIC-circuitlay-outs te ontwerpen, hebben we mesh-knooppunten gebruikt en deze gepositioneerd binnen de PIC-eenheidscellen gevormd door een regelmatig raster van elektrische contactpunten (“bondpads”). Figuur 6 illustreert een voorbeeld van de plaatsing van de knooppunten in de PIC-cellen voor het kleinschalige MEMS-gebaseerde FP-PIC-circuit.



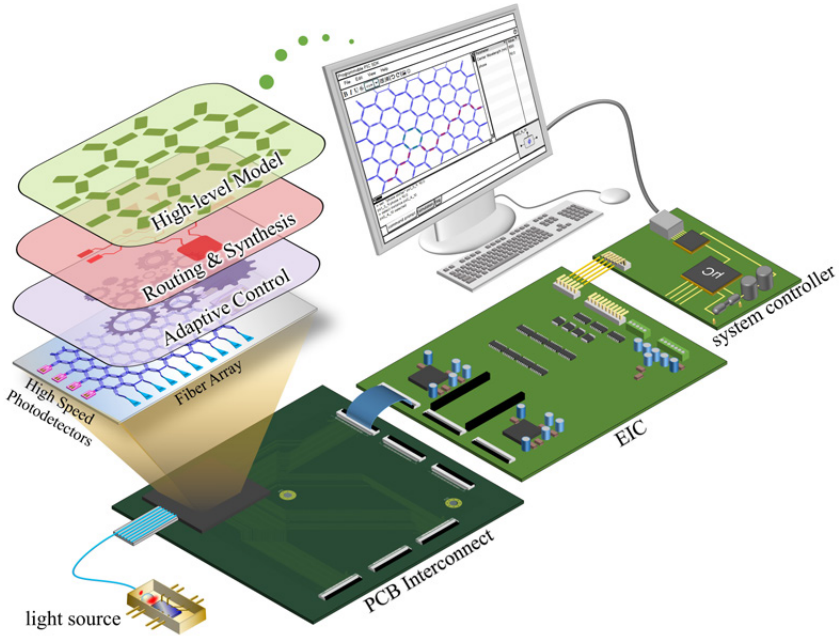
Figuur 6: Implementatie van de knooppunten in de PIC cellen voor de kleinschalig MEMS-gebaseerde FP-PIC.

8 Van Chip tot Systeem

De technologie van het deze multifunctionele programmeerbare fotonische circuits (FP-PIC) is een complex, meerlagig systeem dat fotonische chips, elektronica, verpakking, controlestrategieën en software omvat (Fig. 7). Deze “stack” moet dynamische optische signaalverwerking en herconfigureerbaarheid ondersteunen. De basislaag is de programmeerbare PIC-chip, die verschillende fotonische componenten integreert en fotodetectoren en modulatoren omvat voor microgolfsignaalverwerking. Hierboven bevinden zich analoge en digitale elektronica, zoals spannings-/stroombronnen en microcontrollers, die precieze controle en herconfiguratie van fotonische schakelingen mogelijk maken.

Verpakking speelt een cruciale rol bij het koppelen van de fotonische chip met elektronische en optische systemen, waarbij duurzame materialen en nauwkeurige

montage vereist zijn. Controle strategieën omvatten lokale en globale algoritmes die gebruikmaken van ingebedde fotodetectoren om optische paden binnen het circuit te monitoren en aan te passen. De softwarelaag, inclusief programmeer- en ontwikkelkits, is essentieel voor het configureren en optimaliseren van de functionaliteit van de chip.



Figuur 7: Van een golfgeleidermesh tot een volledig programeerbaar fotonisch systeem. Het golfgeleidermesh is geconnecteerd met lasers op de chip, hogesnelheidsmodulatoren en -detectoren, laag-verlies vertraaglijnen en fotodiodes voor monitoring. Deze chip wordt gekoppeld met een rij optische vezels, elektronische aansturing en circuits om data uit te lezen met een digitale controller, zoals een FPGA. De gebruiker kan dan interageren met de chip via meerdere lagen van aanstuursoftware.

Om nauwkeurigheid te waarborgen, heeft het systeem kalibratiegegevens nodig voor componenten en feedbackcontrole. Op hogere niveaus behandelen programmeeralgoritmes en geautomatiseerde methoden routering, foutcorrectie en verliesreductie. Technieken zoals grafische representatie, machine learning en periodieke kalibratie helpen de prestaties van het circuit te optimaliseren, terwijl er uitdagingen blijven bij het stabiliseren van circuits in dynamische omgevingen en het configureren van complexe functionaliteiten zoals schakelmatrices en filters.

8.1 Verpakking en Fabricageproces

Het fabricageproces begint met wafer-schaalfabricage met behulp van IMEC's iSiPP50G-proces, gevolgd door MEMS-nabewerking en wafer-niveau hermetische afdichting. Na het afdichten worden de wafers in chips verdeeld en gaan ze door de verpakkingsstroom.

Om de fotonische chip te koppelen, met de verzegelde circuits, aan de EIC-borden, gebruiken we interposers en PCB-interconnects. De fotonische chip wordt ondersteboven bevestigd op de interposer met een "flip-chip" proces. De interposer wordt vervolgens op een PCB-interconnect gemonteerd. De PCB-interconnects zijn verbonden met de EIC-borden via flexkabels.

We hebben twee soorten interposers gebruikt om het dichte raster van elektrische bondpads op de fotonische chip uit te waaieren naar een gedrukt schakelbord (PCB), dat verbonden is met de EIC-elektronicaborden via flexconnectoren. Hier volgt een korte toelichting op de interposerbenaderingen:

- **Meerlaagse interposer met hoge dichtheid:** Bij deze methode hoeft slechts één interposer te worden ontworpen. Deze waaiert alle beschikbare elektrische verbindingen op de fotonische chip uit. Dit is een generieke benadering die kan worden gebruikt voor zeer grote circuits. In feite hebben we op elke MORPHIC-chip verschillende circuits geïmplementeerd, en deze methode stelt ons in staat om met één enkele interposer te werken, ongeacht het type en de grootte van de circuits op de chips.
- **Enkellaagse interposer:** In deze aanpak breekt de interposer slechts enkele geselecteerde circuits uit, maar dit is beperkt tot circuits met minder dan 200 verbindingen vanwege de lijndichtheidsbeperkingen op de enkellaagse interposer.

8.2 Circuitaansturing

Onze elektronische besturingseenheden bestaan uit een EIC-bord ontwikkeld door Tyndall, samen met een BeagleBone die dient als digitale controle-eenheid en interface naar het computernetwerk. Het EIC-bord is ontworpen als een flexibele en efficiënte oplossing voor de verschillende schakelingen geïntegreerd in de MORPHIC-chips. Het beschikt over 64 hoogspannings-DAC-uitgangen voor het aansturen van MEMS-bouwblokken en 32 fotodiodeleesinvoeren verbonden met TIAs en ADC's. Deze modulaire aanpak stelt ons in staat om de besturingscapaciteit van de herconfigureerbare PIC's naar behoefte uit te breiden door eenvoudig extra EIC-borden toe te voegen.

Elk EIC-bord wordt bestuurd door een BeagleBone, die via een SPI-interface en GPIO-aansluitingen communiceert met de DAC's en ADC's. De BeagleBone draait

de software die communiceert met het EIC-bord en beheert de regellussen, waarbij de MEMS-aanstuursignalen worden aangepast op basis van fotodiodelezingen. Alle BeagleBone microcontrollers zijn verbonden met een hoofd-pc via een Ethernet-verbinding. Tyndall heeft een low-level programmerings-interface ontwikkeld die zowel zelfstandige opdrachtregelinterface als via een Python-bibliotheek kan functioneren.

8.3 Softwareraamwerk

Een aanzienlijk deel van mijn doctoraatswerk was gewijd aan de ontwikkeling van een Python-gebaseerd softwareraamwerk voor het beschrijven, beheren, programmeren en simuleren van grootschalige fotonische geïntegreerde circuits, met een speciale focus op programmeerbare golfgeleider-meshes. De software maakt niet alleen het ontwerp en de simulatie van een programmeerbare schakeling mogelijk, maar is ook geïntereerd met configuratiealgoritmen, zoals graafgebaseerde routering. Aan de hardware-kant kan het raamwerk ook de verbindingslijsten van de chipverpakking importeren, waarbij wordt bijgehouden hoe de fotonische chip is verbonden met de interposer, interconnect-PCB en de besturingskanalen van elk EIC-bord. Als gevolg hiervan zal het sturen van een afstemmingsopdracht naar een specifieke faseverschuiver automatisch het juiste besturingskanaal op een van de verbonden EIC-borden richten.

Het raamwerk faciliteert de snelle constructie van reguliere golfgeleider-meshes met behulp van verschillende bouwstenen. Het genereert de noodzakelijke graafstructuur voor circuitlayoutontwerp, compatibel met IPKISS-datastructuren om circuitcomponenten te koppelen. Het maakt ook de evaluatie van belangrijke prestatie-indicatoren en mesh-statistieken mogelijk, zoals verwachte verliezen tijdens doorgang, potentiële parasitaire effecten door onvolmaakte bouwblokken of aansturing, en de mogelijke functies die kunnen worden geïmplementeerd. Bovendien integreert het raamwerk ook routerings- en synthesalgoritmen voor een alomvattende modellering.

We hebben het raamwerk specifiek gebruikt om de FP-PIC-demonstrator zo te dimensioneren dat deze effectief de specifieke toepassingen uit het MORPHIC-project, zoals schakelaarmatrices, beamformers en microgolfcircuits, kon implementeren. Een compromis dat we analyseerden was de keuze tussen een mesh met faseverschuivers en één zonder. Het toevoegen van faseverschuivers aan het golfgeleidermesh verhoogt de functionaliteit, waardoor de implementatie van golflengtefilters mogelijk wordt. Echter, een mesh zonder faseverschuivers is beperkt tot routering en lichtdistributie, maar vermindert wel het aantal actuatoren, waardoor er meer meshcellen in een gegeven voetafdruk kunnen geplaatst worden, en ook de algehele optische verlies wordt teruggeschroefd. De grootschalige FP-PIC integreert een mesh die zowel fasegevoelige als fase-onafhankelijke gebieden combineert.

De belangrijkste externe bibliotheken die in ons framework worden gebruikt,

zijn Graphspy (een bibliotheek voor op grafieken gebaseerde algoritmen om padroutering voor enkelvoudige en meervoudige paden te implementeren) ontwikkeld door X. Chen in onze groep, en de IPKISS-bibliotheken door Luceda Photonics voor circuitsimulaties en lay-outontwerp.

9 Karakterisering

De karakterisering van de FP-PIC-schakelingen werd uitgevoerd op zowel verpakte als niet-verpakte chips, die we onderverdelen in de volgende categorieën:

- **Onbewerkte Chips:** Volledige siliciumfotonische chips, maar waarbij de MEMS-apparaten niet zijn losgemaakt (en dus niet kunnen bewegen).
- **Geëtste Chips:** MEMS-componenten zijn losgemaakt en de verbindende golfgeleiders zijn vrij opgehangen.
- **Verzegelde Chips:** Op deze chips zijn de MEMS-bouwblokken hermetisch ingesloten, nadat en verzegelingsproces op wafer-scahll is uitgevoerd. Dit beschermt de MEMS-structuren tegen omgevingsinvloeden en tegen risico's bij hantering.
- **Verpakte Chips:** De verzegelde chips die door Tyndall zijn verpakt en naar ons zijn teruggestuurd voor karakterisatie.

Dientengevolge hebben we verschillende procedures opgesteld:

- Passieve meting van onbewerkte/geëtste/verzegelde chips,
- Passieve meting van de verpakte chips,
- Activeringsmeting van niet-verpakte chips, en
- Activeringsmeting van verpakte chips.

Veel metingen bleken niet succesvol. Tijdens het testen werd een grootschalige instorting van de MEMS-bouwblokken waargenomen in alle circuits, wat leidde tot significante verliezen van licht in het siliciumsubstraat door ingestorte en geblokkeerde golfgeleiders. Dit veranderde het testen in een debug-operatie om de omvang van de instorting in te schatten, te bepalen of er nog paden waren die licht doorlieten, en te identificeren welke MEMS-actuatoren nog functioneel waren. Vanwege de verzegelde verpakking van de demonstratorchips was visuele inspectie niet mogelijk. Verschillende sondeertechnieken, waaronder reflectiemetingen (OFDR), werden gebruikt om het probleem te onderzoeken, en er werd vastgesteld dat de instorting waarschijnlijk het resultaat was van factoren zoals elektrostatische ontlading (ESD), ultrasonische trillingen en substraataarding, eerder dan van de thermische processen tijdens de montage.

10 Conclusie

In deze scriptie presenteren we het circuitontwerp en de meshanalyse van zowel kleinschalige als grootschalige multifunctionele programmeerbare fotonische circuits (FP-PIC's) die gebruik maken van MEMS-technologie. We bespreken uitgebreid de bijbehorende systeemarchitectuur en -componenten, evenals de fabricatie- en verpakkingsprocessen. Het ontwikkelde softwareraamwerk voor het besturen van het elektro-optische systeem in programmeerbare MEMS-gebaseerde FP-PIC's wordt in detail besproken. Er werden verschillende metingen uitgevoerd om zowel verpakte als niet-verpakte chips te karakteriseren, waarbij een hoge uitvalratio in de MEMS-circuits aan het licht kwam ten gevolge van componentinstorting. We bespreken ook het debug-proces om potentiële oorzaken van de MEMS-instorting te identificeren.

Summary

1 Silicon Photonics

The integration density and industrial momentum of silicon photonics has made it an ideal platform for photonic integrated circuits (PICs). Silicon PICs have found their way into biomedical and environmental sensors, LiDAR engines, computing accelerators and quantum information systems. Leveraging the high refractive index contrast and compatibility with advanced CMOS manufacturing processes, we have observed consistent progress from small-scale integration (SSI) to very-large-scale integration (VLSI), with demonstrations of circuits consisting several hundred components [1]. Although some of the functions such as low-power electro-optic phase shifters, integrated light sources [1], or optical isolators have not been offered on any of the silicon photonics platforms, there are tremendous efforts to improve weaknesses of silicon photonics.

2 Programmable Photonic Integrated Circuits

A recently developed category in photonic integrated circuits (PICs) is programmable photonics [11–20]. Programmable PICs are photonic circuits made of waveguides meshes connecting tunable components such as couplers and phase shifters, where couplers' main task is to split the power of propagating light in one or two different paths and phase shifters change the attributed phase of the propagating light. By effectively interfacing tunable optical components on the photonic chip with electronic hardware, we can transmit tuning electrical signals from electronic control units to the chip via commands generated through software interfaces. As a result, we will be able to control light interferences on the chip and create various mathematical tasks such as linear matrix operations [11, 12, 21] or implement desired functions such wavelength filters [17].

The motivation behind developing programmable photonics comes form several properties of these circuits. The *programmability* of such circuits can increase yield and allows for dedicated function implementation. In fact, by fine-tuning each

[1] Some silicon platforms offer some form of light source integration - Openlight, Tower, INTEL.

component on the chip, imperfections resulting from fabrication variations can be corrected, enhancing the probability of the chip functioning properly. Additionally, programmable designs such as switch matrices, photonic accelerators for matrix operations, or quantum-optic circuits can be created for multiple applications, rather than being limited to just one, as is typically the case today.

Another key driver for programmable photonics is the rise of general-purpose Field Programmable Photonic Integrated Circuits [2] (FP-PICs), which are designed as versatile chips capable of being programmed for various functions [22]. FP-PICs represent a significant advancement in the adoption of photonic circuits, as they enable prototyping cycles similar to those of electronic FPGAs. These circuits extend beyond photonic chips alone, requiring the integration of electronics, packaging technologies, and a software programming interface (like other programmable circuits).

3 General Purpose Field Programmable Photonic Integrated Circuits (FP-PICs)

Programmable circuits are categorized in two groups based on their mesh (which is the network of waveguides connecting circuit's tunable blocks) architectures. These two mesh types are "forward-only" and "circulating". In forward-only meshes light signal propagates from the input to the output without any loops or feedback. While, in circulating-meshes, light signal can circulate within the circuit in any direction using loops.

FP-PICs are built based on circulating meshes, which can vary in topology, such as rectangular or hexagonal, and in shape, like rectangular, radial, or parallelogram. For instance, the mesh illustrated in Fig. 1a is a 7-cell hexagonal mesh with a radial configuration. Another key concept in programmable PICs is **2×2 optical gates** (Fig. 1b) which are needed to configure the connectivity and interferences in the waveguide mesh. They can be implemented as tunable directional couplers [2, 23] or as Mach-Zehnder interferometers with phase shifters [24]. Optical gates allow us to mix light from two input waveguides into two output waveguides and control the phase delay between them. In addition to optical gates, **monitors** are key elements within the programmable circuit mesh and should be strategically distributed throughout it. In fact, they are essential for maintaining the circuit's stability in its programmed state. These monitors track optical power (and phase) within the circuit mesh without introducing significant optical loss [25, 26].

[2] Multipurpose programmable photonic processors, or field-programmable photonic gate arrays (FPPGA)

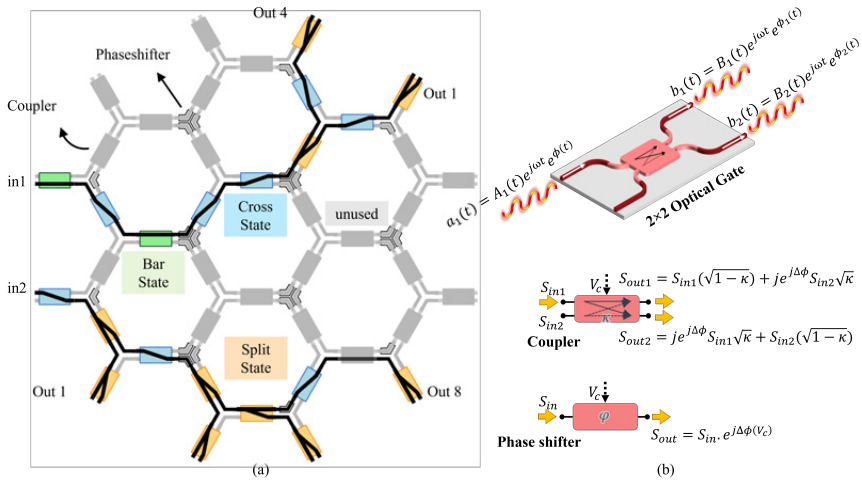


Figure 1: a) schematic of a 7-cell hexagonal mesh, where two different functions are implemented: a 1×6 tap-based beamformer and a 1×4 tree-based beamformer b) 2×2 optical gate, and constructing coupler and phase shifter.

4 Challenges in Scaling Programmable Photonics

An ideal FP-PIC would have a very large size and a vast number of optical gates, allowing for extensive capabilities. However, in reality, we face practical limitations that constrain the size and diminish the performance of our circuits. One of the key questions in the field of FP-PICs is how to scale up the circuits and what feasible solutions exist to overcome limiting factors. Optical loss, power consumption of the circuit components, and packaging limitations are among the key parameters in realization of programmable PICs, especially when we are interested to scale up the circuits.

For silicon PICs, researchers have successfully reduced waveguide propagation loss from $20.0\,\mathrm{dB/cm}$ in early demonstrations to $0.5\,\mathrm{dB/cm}$ in more recent developments [27, 28]. This progress clearly reflects advancements in the fabrication process. Additionally, novel topologies and design strategies can be employed to implement optimized geometries, complementing other process-related improvements. For example, a straight waveguide can be smoothly transitioned into a circular bend using Spline transition approach or Euler bends, which minimizes mode mismatch without significantly increasing the footprint or introducing a loss penalty [29]. In addition to waveguides, grating couplers (which serve as optical interfaces for transmitting and receiving light signals to and from the photonic chip) and optical gates should also be optimized to minimize optical losses of the circuits.

One of the main challenges in FP-PICs is developing a broadband, low-loss, and

compact actuation mechanism. Currently, the most common method is the thermo-optic effect used in heater-based actuators. In fact, heater-based phase shifters offer relatively short optical lengths ($<50\,\mu\mathrm{m}$) and are compatible with monolithic integration in silicon photonic foundries, maintaining low optical losses. However their power consumption is at least $1\,\mathrm{mW}$ per device (often $5\,\mathrm{mW}$-$20\,\mathrm{mW}$).

MEMS-based actuation provides an alternative method to reduce excessive power consumption and high optical losses in individual devices, enabling the scaling of device density in large-scale photonic integrated circuits. This approach requires demonstrating MEMS compatibility with established foundry platforms and integrating them with high-performance standard components. Additionally, MEMS actuators with superior performance compared to other types are essential.

5 The MORPHIC Project

The MORPHIC [3] project (2018-2021) was a European Horizon 2020 initiative aimed at integrating MEMS actuators into silicon photonics to develop Field-Programmable Photonic Integrated Circuits (FP-PICs) operating in c-band ($1.55\,\mu\mathrm{m}$). The project addressed key limitations in silicon photonics, such as variability and high power consumption, by leveraging low-power, non-volatile MEMS actuators, which offer better performance than traditional heater-based tuning. Significant progress was made in integrating MEMS into the IMEC iSiPP50G platform without affecting existing functionalities, as well as in developing compact MEMS-based actuators, wafer-level sealing, and high-density interposers for electrical connections.

In this project, our group was responsible for project coordination, designing and testing small- and large-scale programmable circuits, characterizing circuit components like waveguides and transitions, planning the layout of the MORPHIC chips, collaborating with partners on system component redesign and modifications, and developing technology eco system in general. My contributions included designing and characterizing FP-PIC circuits, developing a software framework for FP-PIC design, analysis, and control, collaborating with Tyndall to create a netlist database for electronic and optical components, testing Electronic Integrated Circuit (EIC) boards, designing small-scale interposers and PCB interconnects, providing feedback to MEMS designers at EPFL and KTH for component optimization, assisting with writing reports and project deliverables, and partially coordinating work package meetings related to my responsibilities.

[3] https://h2020morphic.eu/

6 Mesh Analysis

To design the optical core of the FP-PIC circuits, we performed various mesh analyses, including loss and scaling evaluations based on the critical parameters and measured performance data of MORPHIC MEMS devices. We, also, studied different mesh shapes such as rectangular and radial. Based on our analysis, rectangular-shaped meshes with hexagonal topology require the smallest footprint (for the same number of mesh cells) to support different Application Specific PICs (ASPICs) like switches and beamformers. While radial-shaped meshes exhibit better loss performance than rectangular ones, they demand significantly larger footprints to accommodate ASPIC circuits, negating their advantage in loss performance.

One of the important studies that we performed was reduction of the mesh parasitics caused by the imperfect couplers and phase shifters. In fact, for the meshes with feedback loops, they create a multitude of secondary and tertiary paths for the light, which can cause unwanted interferences and resonances and, thus, affect the frequency response of circuit [30]. Using Monte Carlo simulations we demonstrated that by proper biasing of the unused couplers we can reduces/eliminate the effects of parasitics. For example, Fig. 2 shows the transmission response of a straight line implemented in a 7-cell hexagonal mesh when the unused couplers (shown in gray) are in *bar* and *cross* state.

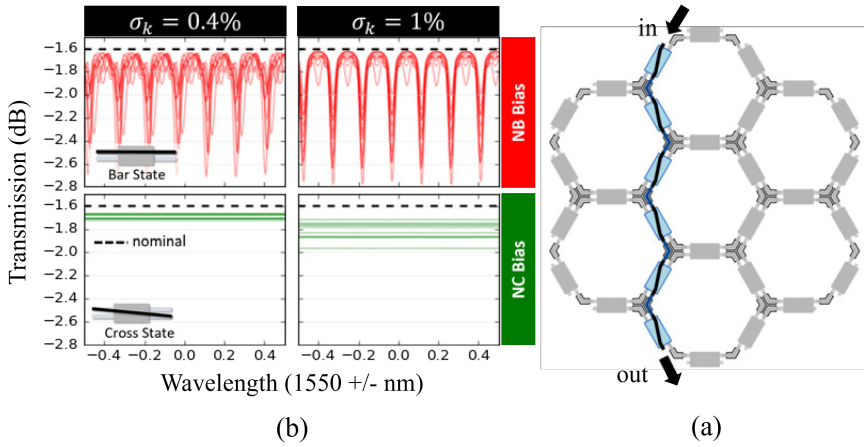


Figure 2: (a) Schematic of a routed path (with $L_{path} = 6L_u$) within a 7-cell mesh. (b) Transmission spectra of the mesh for two types of biasing: normal bar (NB), where unused couplers are biased in the bar state (red curves), and normal cross (NC), where unused couplers are biased in the cross state (green curves). The results are plotted for $\sigma_\kappa = 0.4\%, 1.0\%$ from left to right.

We developed a modeling scheme to assess the errors in programmable photonic

circuits due to imperfections in the phase shifters and tunable couplers. These can have many origins, but one we considered in particular was the discretization error induced by the digital-to-analog converter (DAC). We studied various architectures for both MEMS-based and heater-based 2×2 tunable MZI-based couplers to decrease digitization errors caused by DACs (Figure 3a). Type A is a MZI with equal arm lengths and loaded with an actuator on one arm. In type B, one of the arms of the MZI has a $\pi/2$ phase delay (quadrature) loaded with actuators on both arms. And, in type C, the phase shifters are used together creating discrete 2D-space for the coupling.

Figure 3b demonstrate the $\kappa_{digital}$ response of the three discussed couplers versus the ideal (desired) couplings for a 4-bits voltage driver. Comparing the curves clearly show that using the MZI in quadrature with co-tuning of the phase shifters (type C) increases the accuracy of the coupling selection. This can be seen by comparing the the maximum step size (σ_{max}, illustrated by the black arrow) for these cases.

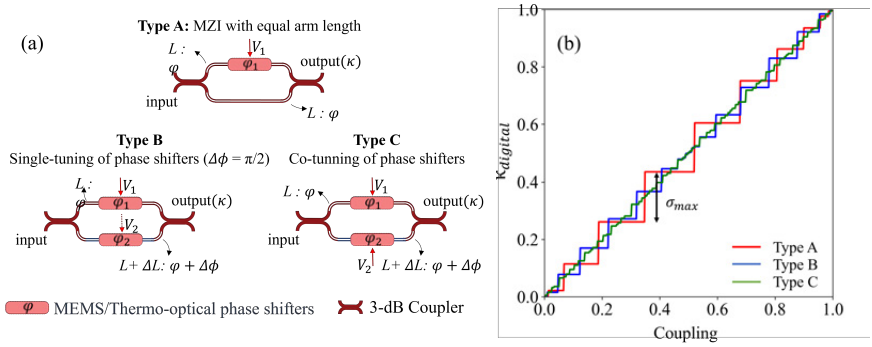


*Figure 3: (a) Tunable 2 × 2 MZI couplers phase shifters. **Type A**: MZI with equal arm lengths is loaded with an actuator on one arm. **Type B**: MZI with a $\pi/2$ phase delay (quadrature) loaded with actuators on both arms. In type B, the coupler is operated in push-pull, where only one of the phase shifters operating at any given time. **Type C**: MZI with a $\pi/2$ phase delay (quadrature) loaded with heaters on both arms. In type C, the phase shifters are used together creating discrete 2D-space for the coupling.*

Additionally, by comparing the number of building blocks (couplers, phase shifters, I/O, and DC connections) per arrangement, analyzing how their numbers scale, and examining how the longest shortest-path loss evolves, we created a comprehensive map for implementing the programmable meshes.

7 Circuit Design

We have implemented two small-scale FP-PIC demonstrators for the MORPHIC project:

- A 7-cell hexagonal mesh actuated by heaters. We also made use of the opportunity to incorporate semiconductor optical amplifiers (SOA) in this mesh, to study how they can help to offset the losses in large programmable meshes.
- A 24-cell hexagonal mesh with MEMS couplers and phase shifters.

Both circuits are illustrated in Fig. 4, are connected to two modulators, 4 pairs of balanced photodiodes, and 8 fiber ports. We have not connected these small circuits to a phased array antenna.

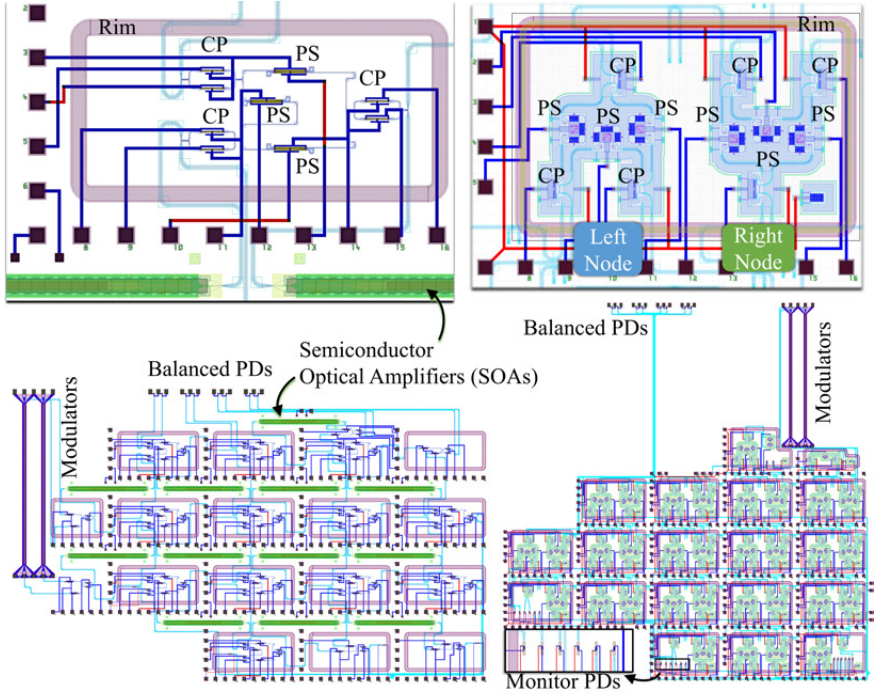


Figure 4: Small-scale FP-PIC designs on MORPHIC RUN2. Left: 7-cell heater-based hexagonal mesh with optional optical amplifiers; right: 24-cell MEMS-based mesh. The PIC unit cell for each circuit is displayed above it.

These designs were followed up by a full demonstrator circuit on the next fabrication run. For our large-scale FP-PIC design ($area > 1\,\mathrm{cm}^2$, with $\simeq 1000$ electri-

cal connections), we implemented a 126-cell FP-PIC using a 14×9 parallelogram-shaped mesh (Fig. 5). The change from a rectangular to a parallelogram shape was made to achieve a cleaner floor plan on the chip, as it could be arranged as a set of rectangular-shaped cells. The fiber inputs (16) and outputs (16) are located on the West and East side, respectively. The North side connects to 4 modulators and 4 balanced photodetectors, and we have also wired up two cavities for the integration of semiconductor optical amplifiers (SOA) using the transfer-printing method [8]. On the South side, we have included additional monitor photodiodes and 16 outputs are connected to a path-balanced phased array antenna.

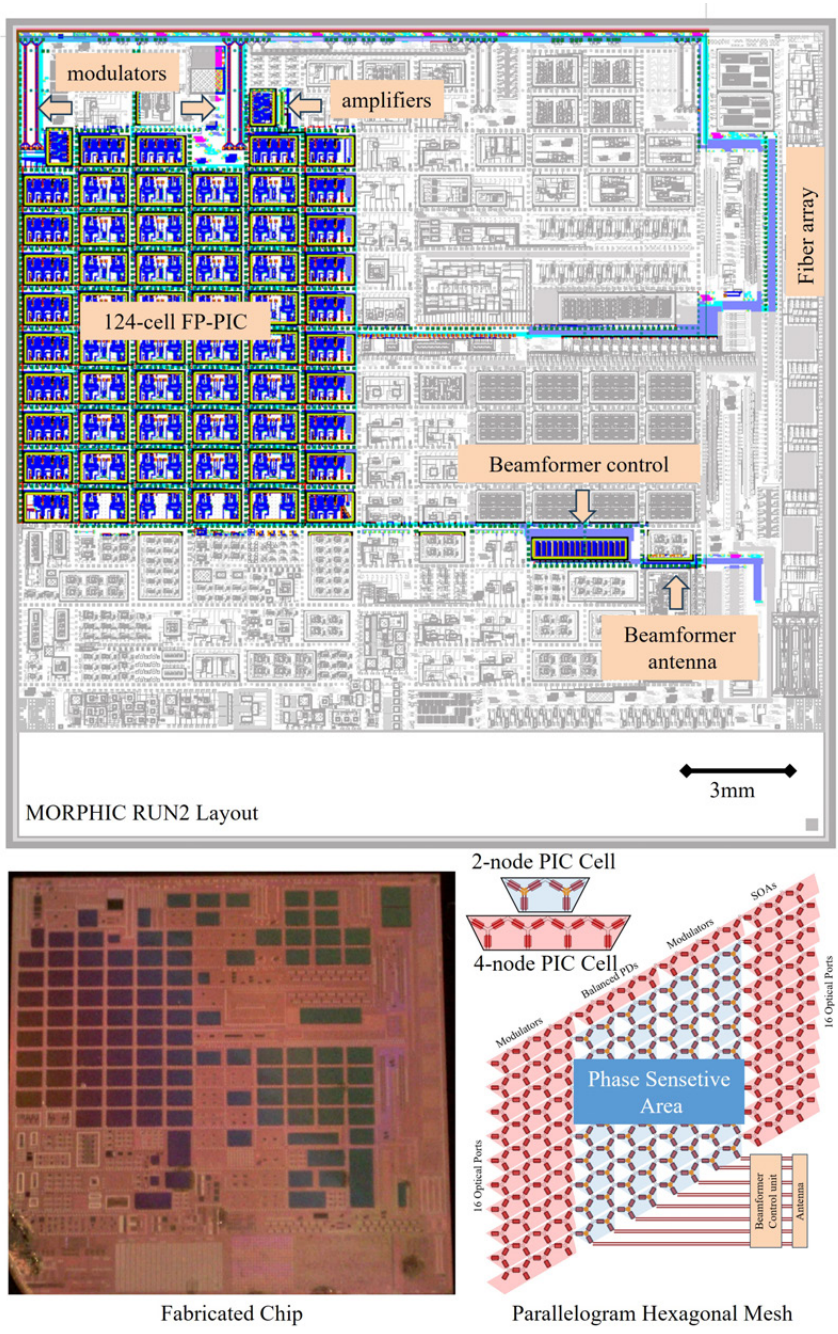


Figure 5: Large-scale FP-PIC with a 126-cell mesh: Layout, schematic, and the fabricated chip.

To design the FP-PIC circuit layouts, we used mesh nodes and positioned them within the PIC unit cells formed by the bondpad grid on the chip. Figure 6 illustrates an example of placement of the nodes in the PIC cells for the small MEMS-based FP-PIC circuit.

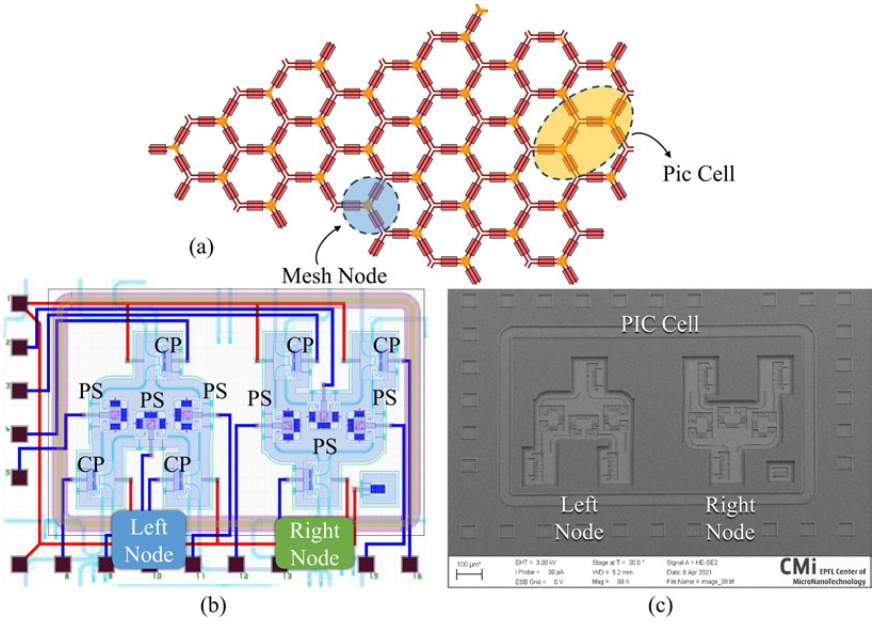


Figure 6: Implementation of the nodes in the PIC cells for the small-scale MEMS-based FP-PIC.

8 From Chip To System

The field-programmable photonic integrated circuit (FP-PIC) technology is a complex, multi-layer system involving photonic chips, electronics, packaging, control strategies, and software (Fig. 7). This ”stack” should support dynamic optical signal processing and reconfigurability. The foundational layer is the programmable PIC chip, which integrates various photonic components and includes photodetectors and modulators for microwave signal processing. Above this, analog and digital electronics, such as voltage/current drivers and microcontrollers, enable precise control and reconfiguration of photonic circuits.

Packaging plays a critical role in interfacing the photonic chip with electronic and optical systems, requiring durable materials and accurate assembly. Control strategies involve local and global algorithms that utilize embedded photodetectors to monitor and adjust optical paths within the circuit. The software layer, including

programming and developer kits, is essential for configuring and optimizing the chip's functionality.

To ensure accuracy, the system requires calibration data for components and feedback control. At higher levels, programming algorithms and automated methods address routing, error compensation, and loss reduction. Techniques like graph representation, machine learning, and periodic calibration help optimize circuit performance, while challenges remain in stabilizing circuits in dynamic environments and configuring complex functionalities like switch matrices and filters.

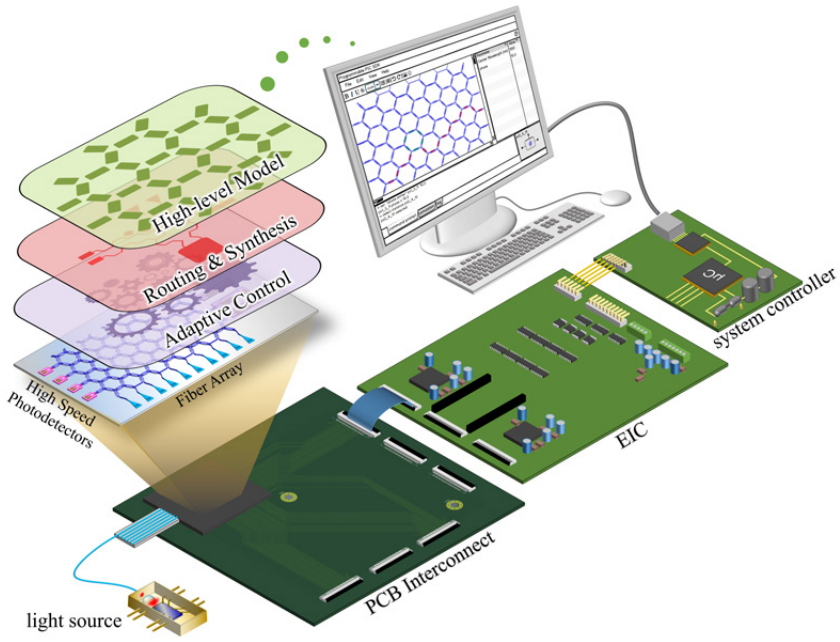


Figure 7: From a waveguide mesh to a full programmable photonic system. The waveguide mesh is connected on chip to lasers, high-speed modulators and detector, low-loss delay lines and monitor diodes. This chip is then interfaced to a fiber array, electronic driver and readout circuitry and a digital controller (for example, an FPGA). The user then interfaces to the chip using multiple layers of programming..

8.1 Packaging and Fabrication Process

The fabrication process begins with wafer-scale processing using IMEC's iSiPP50G process, followed by MEMS post-processing and wafer-level sealing. After sealing, the wafers are diced and proceed through the packaging flow.

To interface the photonic chip, with sealed circuits, the EIC boards, we use interposers and PCB interconnects. The photonic chip is flip-chipped on top of the

interposer, which is then mounted on a PCB interconnect. The PCB interconnects are connected to the EIC boards via flex cables.

We utilized two types of interposers to fan out the dense grid of electrical bondpads on the photonic chip to a printed circuit board, which connects to the EIC electronics boards via flex connectors. Here is a brief elaboration of interposer approaches:

- **High-density multi-layer interposer**: In this method, only one interposer needs to be designed and it breaks out all available electrical connections on the photonic chip. This is a generic approach that can be used for very large circuits. In fact, on each MORPHIC chip we have implemented several circuits, and this method enable us to have a single interposer regardless of the type and size of the circuits on the chips.
- **Single-layer interposer:** In this approach, the interposer breaks out selected circuits, but it is limited to circuits with fewer than 200 connections due to the line density constraints on the single-layer interposer.

8.2 Circuit Control

Our electronic control units consist of an EIC board developed by Tyndall, along with a BeagleBone serving as a digital controller and interface to the network. The EIC board is designed as a flexible and efficient solution for the various circuits integrated into the MORPHIC chips. It features 64 high-voltage DAC outputs for driving MEMS and 32 photodiode readout inputs connected to TIAs and ADCs. This modular approach enables us to expand the control capacity of the reconfigurable PICs as needed by simply adding additional EIC boards.

Each EIC board is controlled by a BeagleBone, which interfaces with the DACs and ADCs via an SPI interface and GPIO connections. The BeagleBone hosts the software that communicates with the EIC board and manages the control loops, adjusting the MEMS driving signals based on photodiode readouts. All BeagleBone microcontrollers are connected to a master PC through an Ethernet connection. Tyndall developed a low-level programming API that operates either as a standalone command-line interface or via a Python library.

8.3 Software Framework

A significant portion of my PhD was devoted to the development of a Python-based software framework to describe, manage, program, and simulate large-scale photonic integrated circuits, with a particular focus on programmable waveguide meshes. The software not only enables the design and simulation of a programmable

circuit but also integrates with configuration algorithms, such as the graph-based routing. On the hardware side, the framework can import packaging netlists, tracking how the photonic chip is connected to the interposer, interconnect PCB, and the control channels of each EIC board. As a result, sending a tuning command to a specific phase shifter will automatically target the appropriate driving channel on one of the connected EIC boards.

The framework facilitates the rapid construction of regular waveguide meshes using a variety of building blocks. It generates the necessary graph data for circuit layout design, compatible with IPKISS data structures to link circuit components. It also enables the evaluation of key performance metrics and mesh statistics, such as expected losses during traversal, potential parasitics due to imperfect building blocks or control, and the possible functions that can be implemented. Additionally, the framework integrates with routing and synthesis algorithms for comprehensive modeling.

We specifically used the framework to dimension the FP-PIC demonstrator so that it could effectively implement the target applications, such as switches, beamformers, and microwave circuits. One trade-off we analyzed was the choice between a mesh with phase control and one without. Adding phase shifters to the waveguide mesh increases functionality, allowing the implementation of wavelength filters. However, a mesh without phase control is limited to routing and light distribution but reduces the number of actuators, allowing for more mesh cells in a given area and lowering overall loss. The large-scale FP-PIC incorporates a mesh that combines both phase-sensitive and phase-insensitive areas.

The key external libraries used in our framework are GraphSpy (a library for graph-based algorithms to implement path routing for single and multiple paths) developed by X. Chen in our group and IPKISS libraries by Luceda Photonics for circuit simulations and layout design.

9 Characterization

Characterization of the FP-PIC circuits was performed on both packaged and non-packaged chips, which we divide into the following categories:

- **Unprocessed Chips**: Full silicon photonic chips, but where the MEMS devices have not been released.
- **Etched Chips**: MEMS components are released and the connecting waveguides are suspended.
- **Sealed Chips:** These chips have undergone a wafer-level sealing process which allows placement of thin caps above the MEMS cavities, which protect

MEMS structures from environmental influences and for handling without the risk of damage.

- **Packaged chips:** The sealed chips which were packaged by Tyndall and shipped back to us for characterization.

Consequently we assembled different measurement setups:

- Passive measurement of unprocessed/etched/sealed chips,
- Passive measurement of the packaged chips,
- Actuation measurement of non-packaged chips, and
- Actuation measurement of packaged chips.

Many measurements were not successful. During testing, a large-scale collapse of MEMS devices was observed in all circuits, causing significant light leakage into the silicon substrate due to collapsed and stuck waveguides. This turned testing into a debugging operation to assess the extent of the collapse, determine if any paths still transmitted light, and identify if the MEMS were still functional. Due to the sealed packaging of the demonstrator chips, visual inspection was not possible. Various probing techniques, including reflection measurements (OFDR), were used to investigate the issue, and it was found that the collapse likely resulted from factors like Electrostatic Discharge (ESD), ultrasonic vibrations, and substrate grounding rather than the thermal processes during assembly.

10 Conclusion

In this thesis, we present the circuit design and mesh analysis of both small- and large-scale Field Programmable Photonic Integrated Circuits using MEMS technology. It elaborates on the corresponding system architecture and components, as well as reviews the fabrication and packaging processes. The developed software framework for controlling the electro-optic system in programmable MEMS-based FP-PICs is discussed in detail. Various measurements were conducted to characterize both packaged and non-packaged chips, revealing a high failure rate in the MEMS circuits due to component collapse. We also discuss the debugging process to identify potential causes of the MEMS collapse.

1 Introduction

1.1 Integrated Silicon Photonics

Photonic integrated circuits (PICs) have grown into an established technology over the past few decades and their complexity has also steadily increased. PICs consist of functional building blocks that manipulate light, connected by waveguides. And PICs are 'chips', i.e. circuits made on a planar substrate. The integration levels in silicon photonics have progressed through several major generations: small-scale integration (SSI), medium-scale integration (MSI), large-scale integration (LSI), and very-large-scale integration (VLSI) (Fig. 1.1). The early generations of the PICs (SSI) could host 1-to-10 components, while, in latest generations (VLSI), prototypes with over 10k components have been demonstrated [1]. Nowadays, we can integrate numerous optical functions such as wavelength filtering, modulation, and photodetection onto a single chip. Similar to electronic integrated circuits (ICs), the functionality of a complex photonic circuit is determined by both the number and connectivity of its elements, as well as the performance of the individual components.

Among technology platforms [31–34] available for fabrication of photonic chips, silicon photonics [31, 33], silicon nitride photonics [34], and indium phosphide PICs [32] are promising to accommodate a large number of building blocks

due to their high refractive index. However, Silicon photonics stands out as the only photonic integration technology capable of supporting the required component density for continued scaling of complexity in PICs. This is due to its high refractive index contrast and compatibility with advanced CMOS manufacturing processes [27–29, 35, 36].

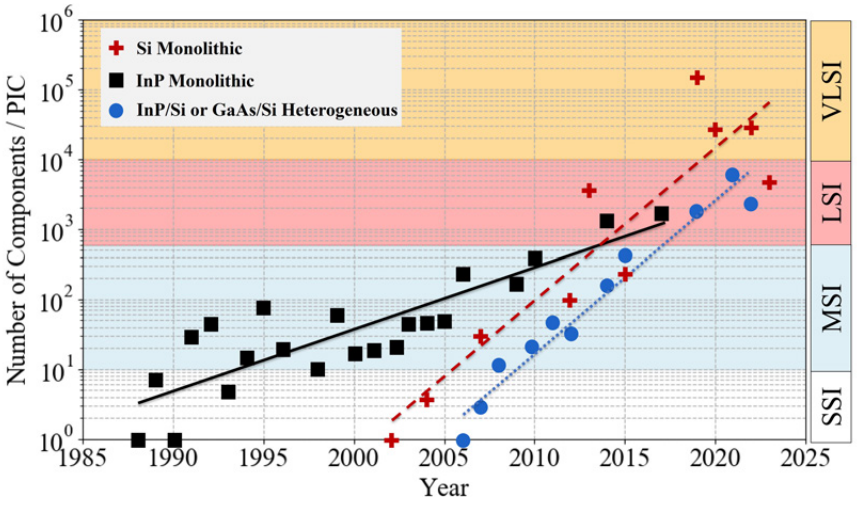


Figure 1.1: Timeline showing the number of components on a silicon photonic integrated circuit (PIC) over different generations: small-scale integration (SSI), medium-scale integration (MSI), large-scale integration (LSI), and very-large-scale integration (VLSI). (This figure is regenerated from [1])

The significant contrast between silicon and its oxide (SiO_2) allows for the confinement of infrared light within sub-micrometre optical waveguides and the ability to bend light with a radius of about $5\mu m$. Silicon photonic circuits are typically implemented on silicon-on-insulator (SOI), which aligns with the fabrication processes used for CMOS electronics, thereby enabling the use of an established high-volume manufacturing base. As a result of these advantages, research in silicon photonics has surged over the past decade, leading to industrial adoption and the emergence of initial products on the market [37]. Currently, this advancement is primarily driven by data center communications and telecommunications [38], but silicon photonics is also demonstrating its potential in (bio)sensing, microwave, and spectroscopy applications [39–41].

Figure 1.2 shows the schematic of both passive and active components which are common in Silicon photonic integrated circuits. Waveguides, both straight and bent, direct light through PICs, while directional couplers enable flexible power

splitting ratios between different waveguides. Grating couplers function as optical inputs and outputs (I/Os) for the PIC, and disk and ring resonator structures provide wavelength-selective filtering capabilities. Active components leverage electrical and optical effects to generate, modulate, and detect light. Light produced by a laser source can be coupled into a PIC through a grating coupler, then routed via waveguides to passive devices or modulators for specific functions such as phase shifting, switching, or variable power coupling. After performing a particular operation, rather than coupling the light back out of the PIC for measurement with a power detector, the analysis can be conducted in the electrical domain using a photodetector.

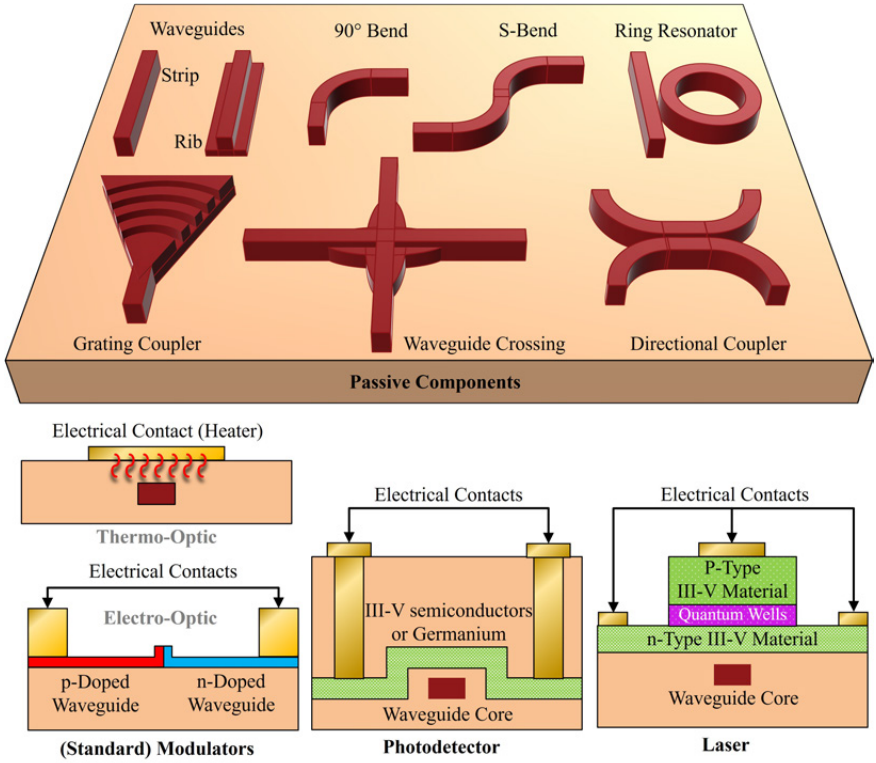


Figure 1.2: Schematic of the common passive and active components of silicon Photonic Integrated Circuits (PICs).

While silicon photonics technology has made impressive progress, it still faces several limitations that must be addressed to achieve widespread adoption and optimal performance. Ongoing research and development efforts are focused on overcoming these challenges, including improving material properties, enhancing device performance, refining integration techniques, and establishing industry

standards. Addressing these limitations will be crucial for the future success and scalability of silicon photonics in various applications, from high-speed communications to advanced computing systems.

1.2 Programmable Photonic Integrated Circuits

Programmable photonic integrated circuits (programmable PICs) are photonic chips that can be reconfigured via software to perform various functions across different applications. The key feature of these circuits is the *reconfigurability for multiple purposes*. Hence, it is important to distinguish between a truly general-purpose (multifunctional) programmable PIC and a tunable ASPIC (Application Specific Photonic Integrated Circuit), where the functionality can be electrically tuned but the circuit is designed for one specific purpose [42]. For example, in a Wavelength Division Multiplexing (WDM) system, a tunable micro ring resonator (MRR) based optical filter can be used to dynamically select specific wavelength channels from a multi-channel optical signal. This tuning can be electrically controlled in real-time, providing the ability to reconfigure the system for different communication needs, without physical modifications. However, this circuit can not act as a beamformer which has a totally different functionality.

In general, programmable PICs feature *waveguide meshes* with tunable couplers and phase shifters, which can be electrically adjusted to define diverse functions and arbitrary connectivity between input and output ports. This allows software-controlled distribution and rerouting of light and, consequently, the real-time manipulation of light on the chip. Such chips can execute various linear operations by creating interferences along different paths and can also define programmable wavelength filters [17], which are crucial components for communication and sensor applications, as well as for manipulating microwave signals in the optical domain [18,43]. As waveguide meshes scale up, their interferences can perform linear optical computations, such as real-time matrix–vector products [11, 12, 21]. These operations are fundamental in quantum information processing [13,44–46], artificial intelligence, and neuromorphic computing [11, 21], where rapid advancements in programmable PIC technologies are happening. Similar to electronics, programmability in PICs allows for (re)configuring functionalities at runtime, thereby reducing economic and technological barriers and providing a pathway to upgradability.

1.2.1 Mesh Architectures

The heart of the optical core of programmable photonic circuits are waveguide meshes which are categorized into two main types: (1) forward-only meshes [11–16] and (2) recirculating meshes [17–20]. *Forward-only meshes* (Fig. 1.3a-b) are typically organized in a linear or grid-like structure where the light signal propagates from the input to the output without any loops or feedback paths. These meshes can be described by a Transmission Matrix (T-matrix) representation, describing the coupling between the input ports and the output ports.. In contrast, *recirculating meshes* (Fig. 1.3c-d) incorporate feedback loops, allowing the light signal to circulate within the circuit creating complex resonant structures. We can use a Scattering Matrix (S-matrix) to mathematically represent these meshes.

The choice between feedforward and recirculating meshes in programmable photonic integrated circuits depends on the specific application requirements. Feedforward meshes are suitable for simpler, low-latency applications where predictability and straightforward design are crucial. These meshes do not allow for the construction of filters, as they are usually designed with balanced path lengths. They have a broadband operation. And, their implementation is mostly for performing linear operations (matrix-vector multiplication, quantum-optic gates, adaptive beam coupling, mode separation). In contrast, recirculating meshes offer greater flexibility and advanced functionalities, making them ideal for complex signal processing tasks, albeit at the cost of increased design complexity and potential stability issues.

1.2.2 2×2 Optical Gates

The key components for waveguide meshes are 2×2 optical gates (Fig. 1.4a), which act as tunable couplers. These gates mix light from two input waveguides into two output waveguides and control the phase delay between them. The minimum components needed to build these meshes are phase shifters and tunable couplers (Fig. 1.4b). By connecting these components, we can reroute light just by switching the optical gates between bar and cross states. The 2×2 optical gates can be constructed by different arrangement of phase shifters on a MZI-based coupler or combining with a 2×2 coupler (Fig. 1.4c). The functionality of these gates extends beyond simple routing. By using partial coupling states, light can be redistributed through multiple paths and recombined, creating a programmable interferometer. This can introduce path delays, creating tunable filters. You can also implement filters using resonators by routing light into loops, enabling wavelength-dependent responses [17].

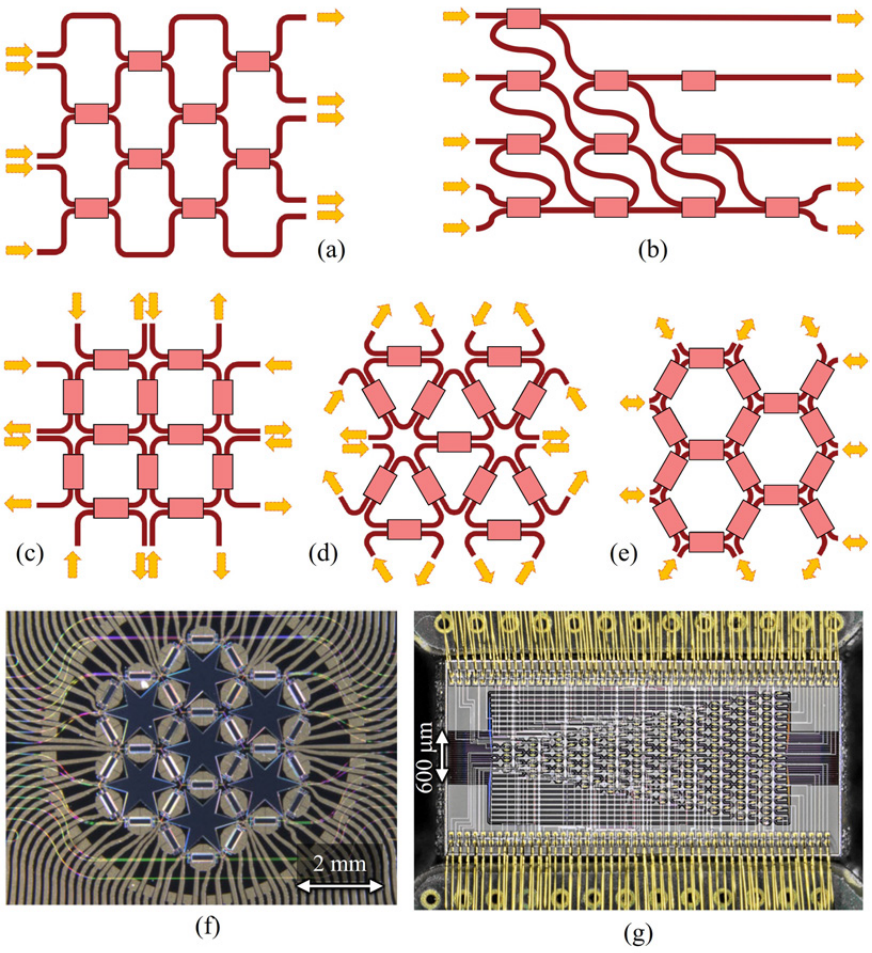


Figure 1.3: Forward-only meshes with a) rectangular and b) triangle unitary architectures. And, recirculating meshes based on c) square, d) triangle, and e) hexagonal cells. As seen, couplers (pink rectangles) are connected with waveguides and yellow arrows show optical inputs and outputs. Demonstration of a f) 7-cells hexagonal recirculating mesh developed by the Capmany group at Universitat Politecnico de Valencia [2], and g) large-scale forward-only mesh with 26 input channels by the Englund group at MIT [3].

1.2.3 Actuators

Optical gates in programmable circuits can be implemented by using *actuators* (or tuners) which are tunable couplers or phase shifters forming the building blocks of the waveguide meshes in programmable circuits as discussed. Ideally, to satisfy requirements of large-scale waveguide meshes, these actuators should have the

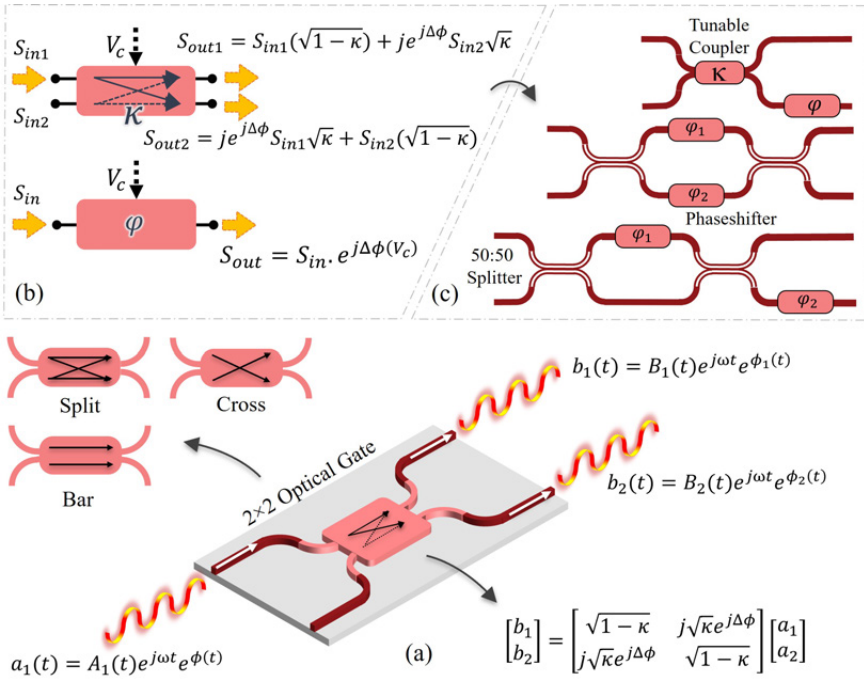


Figure 1.4: a) 2×2 optical gates can mix two input optical waves (a_1 and a_2) or split optical waves coming from one of the input waveguides (a_1 or a_2) by controlling both the power coupling κ and the phase delay $\Delta\phi$ of the output waves (b_1 and b_2). For simplicity, we assume that the gates have no optical loss. b) These gates can be constructed using couplers and phase shifters and be tuned between 'cross' and 'bar' states. c) They can be implemented as a MZI with two phase shifters or tunable coupler with an additional phase shifter. In all cases we will have a circuit with two degrees of freedom.

following features:

- Compact in size (with short optical length)
- Low optical insertion losses
- Broadband response
- Low electrical power consumption (preferably with CMOS-compatible voltages (< 5V))
- Short response time

Figure 1.5 illustrates schematics of various actuation mechanisms for the building blocks of programmable photonics. Here, we briefly elaborate these tuning

mechanisms for phase shifters which can be used to build couplers and switches as well.

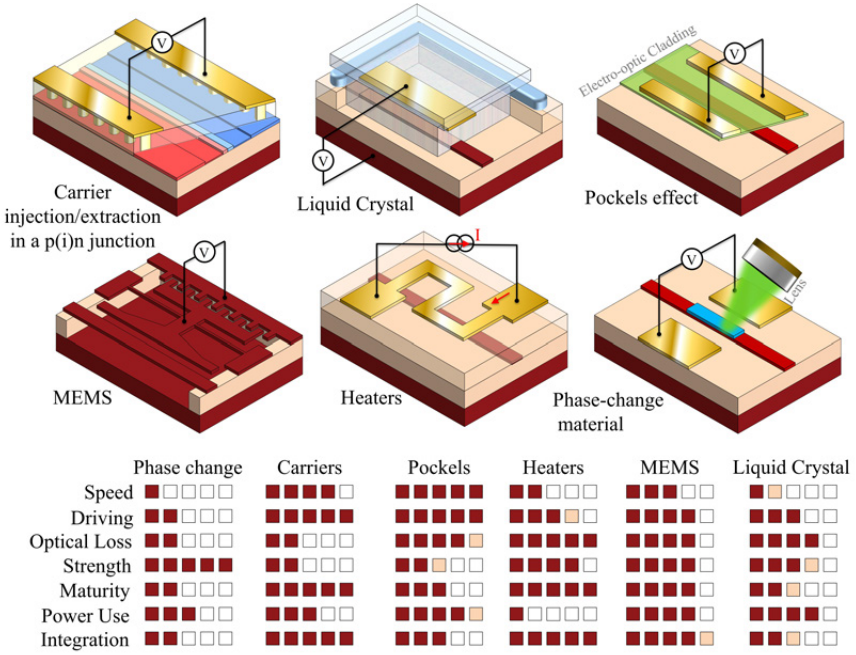


Figure 1.5: Semi-quantitative comparison for different available actuation mechanisms for silicon photonics. (This is figure is regenerated from [4])

In pockels effect, the application of an electric field induces a change in the refractive index proportional to the field strength. This refractive index change directly alters the phase of light propagating through the material, enabling precise phase control. Liquid crystal based devices leverage the electro-optic properties of liquid crystals, where the refractive index can be tuned by applying an electric field, causing the liquid crystal molecules to reorient. In Phase-change materials the refractive index changes when transitioning between distinct solid-state phases (typically amorphous and crystalline). This refractive index change happens when they are exposed to thermal, optical, or electrical stimuli. Carrier injection/extraction in p-n junctions leverages the plasma dispersion effect, where changes in free carrier concentration (electrons and holes) alter the refractive index of the semiconductor material.

Thermo-optic actuators (heaters) [47–50] are a commonly used tuning mechanism in silicon photonics because of their straightforward fabrication process (they can be implemented in most platforms as an electrical resistor). These actuators employ a heater placed in proximity to the optical waveguide, utilizing the tem-

perature dependence of the refractive index to achieve phase shifts. Despite their simplicity and effectiveness, integrating large numbers of thermo-optic tuners is challenging. They are power-hungry, require adequate spacing to prevent thermal cross-talk, and have relatively slow response times ranging from microseconds to milliseconds [51]. In fact, most efficient thermo-optic tuners face scalability issues with power dissipation and thermal management, highlighting the need for more efficient alternatives [42, 52–54]. Examples of thermo-optic actuators are silicon photonic heaters with $P_\pi \approx 1 - 5mW$ for undercut heaters and $P_\pi \approx 15 - 20mW$ for regular heaters [50, 55, 56].

An alternative to thermo-optic actuators are fast tuning mechanisms such as carrier-based plasma dispersion and linear electro-optic (EO) phase shifting (or Pockels). However, they have their own limitations. The first mechanism suffers from high insertion losses and long optical lengths [53]. Additionally, in this tuning mechanism, the basic 'unitary' function of the optical gate will be destroyed because of dependence of loss on the induced phase shift [42]. The second mechanism also suffers from large foot prints; for example, phase shifters using Lithium Niobate will be several millimeters long which depends on the driving voltage. Additionally, integration of silicon with electro-optic materials (Lithium Niobate, Barium Titanate (BTO) or polymers) is challenging; in fact, the heterogeneous integration poses significant challenges due to the complexity of incorporating these materials with existing silicon photonics infrastructure.

There is an ongoing research to propose improved or alternative approaches for actuation mechanisms for the phase shifters. These efforts include piezoactuators [57, 58], liquid crystals [59, 60], and MEMS [61]. Moreover, phase-change materials [62, 63], memristors [64], and mechanically latched MEMS [61] can act as non-volatile actuators where they can maintain their state without an 'always-on' control signal. In general, the 2×2 gates containing these actuators are the main source of loss in the circuit, with values around $0.05\,\mathrm{dB}$-$0.5\,\mathrm{dB}$ per gate [42]. An important aspect in scaling up programmable photonic integrated circuits is reducing these loss values, and, fortunately, this is happening by technological advancements and more optimized component designs.

1.2.4 Configuring A Recirculating Mesh

In this work we mostly look at recirculating meshes, and in particular hexagonal meshes. Figure 1.6 shows some basic configuration examples for a recirculating

7-cell hexagonal mesh. As seen, by setting the coupling value of the couplers (changing their state) we can manipulate light propagation within the mesh and implement different functionalities. The default state of the mesh is in cross which means all the (gray) couplers are in cross state. Blue and green colors means that selected coupler is actively configured in cross or bar state, respectively. And, the split state is shown by the orange color in which the coupling value of the couplers is between 0.0 and 1.0. For example, Fig. 1.6a and Fig. 1.6b show the implementation of a ring resonator and a Mach-Zehnder Interferometer (MZI), where by changing the state of only three couplers we can switch between these two circuits. In Fig. 1.6c, we have shown the implementation of two delay lines (single paths) with different lengths.

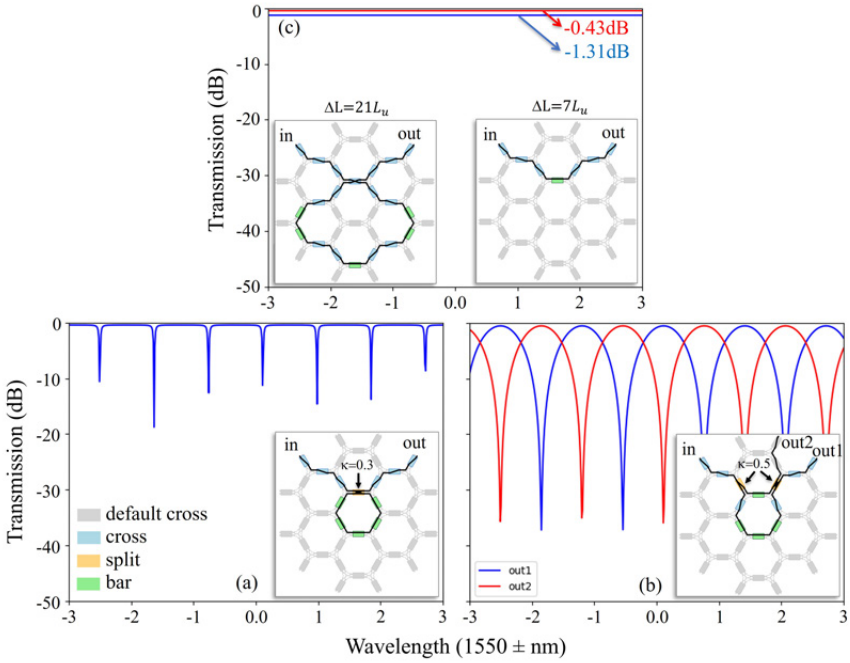


Figure 1.6: Basic configuration examples for a recirculating 7-cell hexagonal mesh: a) Ring resonator, b) Mach-Zehnder Interferometer (MZI), and c) Two single paths with different length acting as delay lines.

1.3 Field-Programmable Photonics Integrated Circuits

Over the past decades, electronics have advanced significantly with innovations such as digital processors, Field-Programmable Gate Arrays (FPGAs), and microcontrollers. Similarly, photonics has made big steps in integrating multiple functions onto a single chip. However, there is a key difference: in electronics, general-purpose programmable devices like microcontrollers and FPGAs emerged early on (in the late 1980s), whereas current photonic circuits are typically application-specific integrated circuits (ASICs) designed for specific functions. Field-Programmable Photonics Integrated Circuits (FP-PICs) [1] are a special type of programmable photonic circuits which are expected to play a role similar to that of FPGAs and microcontrollers in electronics, fostering development and widespread industrial use.

As a short comparison of the electronic FPGAs and FP-PICs, we should mention that FPGAs are electronic devices with reconfigurable logic blocks interconnected through programmable routing, allowing them to execute digital logic, DSP, and other computational tasks. FP-PICs, on the other hand, are programmable integrated circuits designed to manipulate light rather than electrical signals. They use components like optical waveguides, modulators, and phase shifters to route, modulate, and process optical signals. In FPGAs, programmability is achieved by reconfiguring the internal logic gates, flip-flops, and I/O connections. The programming process is often done using hardware description languages (HDLs), like VHDL or Verilog. FP-PICs achieve programmability by using tunable photonic components like phase shifters, couplers, and thermo-optic or electro-optic modulators connected by the mesh of waveguides. Programming FP-PICs usually requires control over physical parameters (such as temperature or voltage) to tune optical paths and functionalities, often using custom software that interacts with these components.

The solution for creating Field-Programmable Photonics Integrated Circuits (FP-PICs) was proposed several years ago through *recirculating waveguide meshes* [13]. And, the Polytechnic University of Valencia demonstrated the first implementation of such a circuit in 2017 with a 7-cell circuit containing 60 tunable phase shifters (fig. 1.3f). This circuit could be configured for 100 different functions. Additionally, researchers at KAIST and DGIST in Korea have developed programmable photonic circuits that feature exceptionally low power consumption and reconfiguration energy, leveraging silicon photonics and microelectromechanical systems (MEMS) technologies. The MEMS-tunable couplers and phase shifters enable precise con-

[1] or Generic Programmable Photonic Integrated Circuits

trol of optical signals, with a single-element standby power consumption of under 10 femtowatts and a maximum reconfiguration energy of just 40 picojoules. These represent the lowest static power and tuning energy reported to date for densely integrated programmable photonic circuits [65].

To make recirculating meshes programmable, we need to combine the actuators (switches, tuners and couplers) with electronic control and feedback loops. This way, custom connectivity can be defined (rerouting, splitting, multi-casting), as well as optical functionality based on interference effects (wavelength filtering, sensing). Combining such programmable waveguide meshes with existing active optical components such as high-speed modulators [11] and photodetectors, a generic photonic chip can be constructed that can be field-programmed to perform a variety of functions. Figure 1.7 shows a conceptual design of a generic programmable photonic circuits. We should note that waveguide meshes accumulate higher optical losses compared to application specific PICs, and the situation get worse when we scale up the programmable circuits. Hence, by placing on-chip amplifiers either inside the waveguide meshes or on their edge we will be able to compensate such losses in large-scale programmable circuits. Additionally, combined with waveguide cavities, on-chip amplifiers also can act as programmable laser. Despite of integration challenges, III-V amplifiers can be integrated to silicon chips using different techniques such as microtransfer printing [8, 66] or bonding [66]. We should mention that up till now, no demonstrations of waveguide meshes with built-in amplifiers have been realized.

The field-programmable PIC is more than just a photonic chip. The developments of the wafer-processing, the photonic building blocks and subcircuits, the reconfigurable circuit meshes, the packaging solution, electronic control circuits, and software framework result in a complex and multi-layer technology stack. Such a photonic-electronic-software stack is needed to control and program the chip's functionality, enabling dynamic optical signal processing and reconfigurability. In our view, this technology stack can have several layers of programmable PIC chip, electronics (analog/digital), packaging, control strategies, programming, and developer kit (Fig. 1.8).

The **programmable PIC chip** is the bottom layer of our technology stack. It is related to the fabrication processes, postprocessings (for example releasing the MEMS structures after receiving the chips from a foundry), wafer level testing, and realization of a fully functional high-density PIC platform supporting not only waveguiding, modulation, photodetection, fibre coupling and tuning but also (ideally) light sources and amplifiers. Such optical chip can also host photodetectors

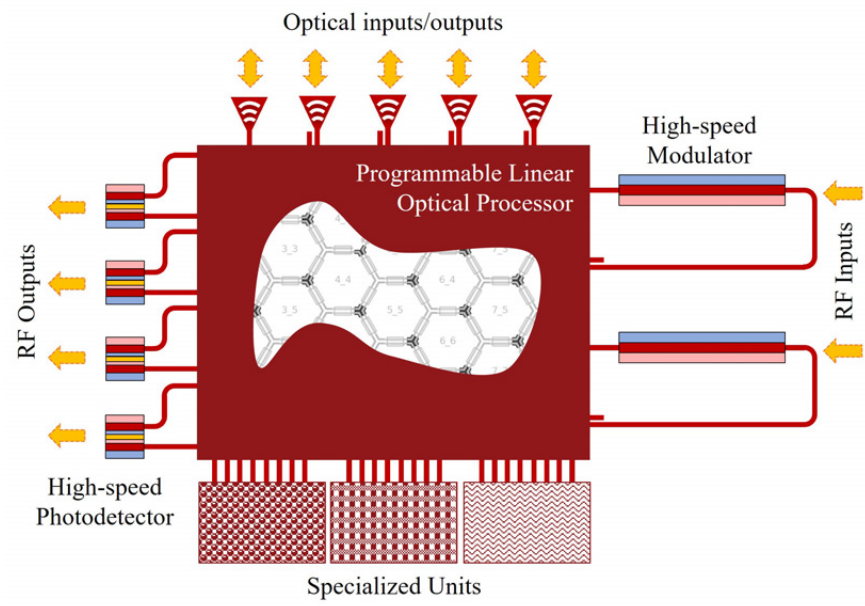


Figure 1.7: Schematic of a Field Programmable Photonic Integrated Circuit (FP-PIC).

and modulators providing the outputs and inputs for microwave signals that will be processed on them [17,18,43,67]. For example, we can use high-speed electro-optic modulators with bandwidths exceeding 50 GHz [5, 68].

The next layers on top of programmable PIC chip are **analog and digital electronics**. An optical phase shifter or tunable coupler makes it possible to adjust or reconfigure a photonic circuit. And, to actuate these components, we need to construct a *modular system* composed of microcontrollers (FPGs), voltage/current drivers, and readout circuits in which a hierarchical digital network can be programmed from a user computer. In such a system, couplers/phase shifters are connected to voltage/current drivers and the photodetectors embedded on the photonic chips are connected to readout subcircuits. In the control architecture, microcontrollers (or FPGs), using command signals, select voltage/current drivers (Digital-to-Analog Converters (DACs)) to send driving signals to actuators or select readouts (Analog-to-Digital Converter (ADCs)) to read the output of photodiodes. In general, digital and analog electronics can be used to implement high-level control loops and fast control loops, respectively.

Moving up in our technology stack, we reach to the **packaging**. In order to

send/receive both electrical and optical signals to/from the programmable photonic chip, it should be properly interfaced with electronic hardware and RF/optical sources and monitors. In fact, the photonic circuit can no longer be considered isolated from its control electronics and software algorithms, and the necessary interfaces to electronic drivers and programming logic should be developed. These interfaces can include high-density fibre interfaces, electrical interposers, and interconnect PCBs which are essential components in the packaging and assembly process. They should be compatible with driving voltages and currents. Moreover, the materials used in the photonic chip must endure the temperature changes during assembly, and the dimensions of both the features and the entire chip must be compatible with the assembly tools. Another important point to consider during packaging is to implement optical and electrical test devices to assist the packaging process and verify the quality of the assembly. For example, we can use shunt waveguides for fiber alignment, and electrical daisy-chains to evaluate the yield of the electrical connections. Tyndall, a world leader in these technologies, has already demonstrated several packaging concepts and examples [69, 70].

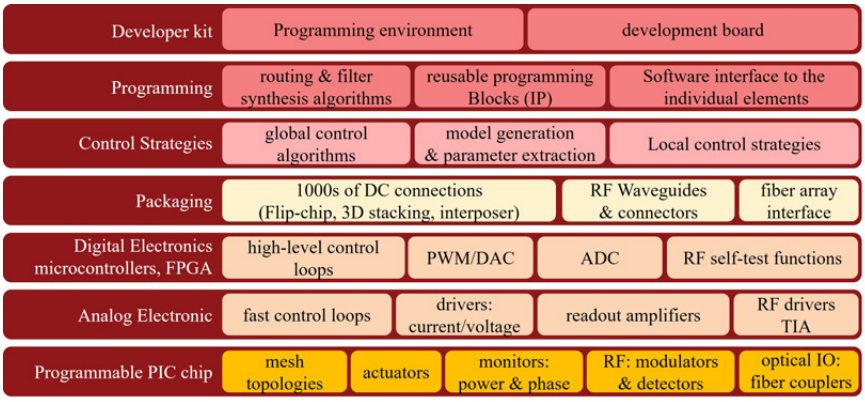


Figure 1.8: Technology stack for field-programmable photonic integrated circuits (FP-PICs).

Control strategies is the next layer in the technology stack with the objective of ensuring accurate and readable actuation of couplers and phase shifters using local and global control algorithms. A large-scale FP-PIC can have thousands of possible optical paths; hence, to control actuators we need to know where the light is on the waveguide mesh of the chip and track its path [71]. This can be achieved by embedding monitor photodetectors either inside [26, 72] or immediately after the optical gates [14, 25]. Combining monitor photodiodes and their electronic readouts with optical gates, we can close the loop with feedback control and build subcircuits that can monitor their own state and adjust their performance. The feedback

loops can be implemented using software or analogue/digital electronics [42]. These subcircuits do not require centralized control system and self-configuration algorithms [12, 73, 74] can be used. A simple example of such *self-configuring subcircuits* is minimizing the light on a photodiode by tuning the phaseshift and coupling ratio [12]. Monitor photodiodes inside self-configuring subcircuits can increase overall loss of the FP-PICs by stealing light from the optical paths of the mesh; to minimize optical losses, such monitors should be made as transparent as possible which can be done by various methods such as using waveguide taps [14, 75] or using surface state absorption in silicon waveguides [25, 72, 76].

To complete our technology stack we need to build software layers controlling various aspects of our programmable chip [77]. We refer to these layers as **programming** and **developer kit** which can be accomplished by a well-designed software framework. Ideally, we are interested to have full control over the actuation of tunable blocks (couplers/phase shifter), to configure the mesh for a specific or multiple functions, to stabilize implemented circuits or desired actuators to their desired states, and to implement optimization methods, routing algorithm, and synthesis techniques. Additionally, developers would require development kits and an application programming interface (APIs) as programming infrastructure facilitating their interaction with photonic/electronic hardware.

To enhance control accuracy of the actuators on the chip circuits, we need calibration data of components and feedback loops (as discussed before). For the actuators, we should collect the actuation curves (e.g., phase shift vs. voltage) of devices to understand the statistical variation. And, for the different photodetectors (e.g. monitor photodetectors), we need to collect the responsivity and dark current. This data not only feeds into the specs for the electronics but also can be used as modifying parameters when mapping desired phaseshift/coupling to actuation signals (voltage/current) applied to the actuators. Moreover, feedback control routines, at the lowest level, can use the readout of the on-chip monitors to stabilize the actuators to their desired state which in some cases can be a simple maximizing or minimizing routines [12].

At a higher level, configuring a mesh with thousands actuators requires programming algorithms, proper mesh representation, and automated methods. The generic programmable chip will not have ideal photonic components because of errors in fabrication process or design cycles. We also may have to compensate errors caused by electronics such as discrete response of the DACs [78]. All these can cause parasitic effects [79–81] on response of the circuits implemented in the mesh. Loss is an important issue in generic programmable photonic chips; imagine

that we are interested to implement several circuits in the mesh at the same time, what would be the ideal place for the circuits to be on the mesh to have lowest possible loss. Or, how we can find shortest path on the mesh with lowest loss? Or, how we can implement several paths on the mesh, and how the autorouting should be done?

Another important question to answer is how to control the circuit mesh during its operation? Considering sensitivity of photonic components to the environmental changes, it is expected that the response of the circuit elements will drift over time. Hence, it is important to have methods to control and stabilize the set points of the actuators. For forward-only meshes some routines have already been described [12]; however, it is a non-trivial unsolved problem in recirculating meshes that also need to have controlled wavelength dependence for filter function. In this case, dithering or pilot tones could allow to disentangle the contributions of individual actuators to an output signal.

The next possible aspect of programming layer is configuring the photonic chip for specific functionality such as switch matrix or optical filter function. Thus, autorouting functions [82] and filter synthesis alongside with necessary tools to visualize and debug the configuration of the chip are important.

A high-level programming of a photonic PIC should make it possible to use and deploy PIC technology without a significant know-how of semiconductor processes or even the internal workings of the photonic components. A good API smoothens the design process, reducing the time and effort required to create and test complex circuits. And, it helps to automate repetitive tasks allowing designers to focus on innovation and optimization. Similar to VHDL for FPGAs in electronics, FP-PICs may need their own descriptive language. Where reusable routines to operate programmable chips can be shared/modified between/by developers and users. In fact, reusable blocks of code can drastically shorten development time and also chips characterization. This can result in a fully operational photonic–electronic systems-on-chip.

1.4 The MORPHIC Project

Most of my PhD research was carried out in the context of the project MORPHIC (Mems-based zerO-power Recofigurable PHotonic ICs) which was a European Horizon 2020 project aiming to enhance a Silicon Photonics platform with MEMS[2] actuators to create a technology platform for generic Field-Programmable Pho-

[2]Micro-Electro-Mechanical Systems (MEMS).

tonic Integrated Circuits (FP-PICs). From 2018 to 2021, this vertically integrated project brought together six partners[3] with expertise in silicon photonics, photonic MEMS, photonic-electronic packaging, circuit design, and various application areas (www.h2020morphic.eu).

MORPHIC has addressed two limitations of today's silicon photonics technology by introducing MEMS into silicon photonics [23, 83]. These limitations which constrain scaling of photonic circuits are: *Variability* and *Power consumption*. In fact, the high contrast of SOI makes the components highly sensitive to nanometer-scale variations. As a result, even the most advanced processes are limited in the number of components a circuit can contain without experiencing mismatched responses due to stochastic variations. Although such variations can be compensated by tuning of the circuits actuators, high power consumption of the thermal actuators limits accommodation of many tuning elements.

Photonic MEMS actuators provide one of the strongest possible electro-optic effects in on-chip technology [84], because they induce a strong local index change by moving a high-index material such as silicon. Moreover, as MEMS work through electrostatic force, they have a very low power consumption. And, by adding mechanical latching mechanisms, MEMS can be made non-volatile and can maintain their state without power. It is worth to mention that non-volatile and electrostatically actuated MEMS have an inherent advantage in terms of power consumption compared to traditional heater-based tuning and switching (nW instead of mW). They also have better optical loss ($<0.5\,\mathrm{dB}$ per actuator) performance compared to charge carriers and even electro-optic materials. This characteristic can be used to considerably improve performance of large-scale and complex PICs.

In MORPHIC, we have overcome the key challenge of integrating MEMS into the IMEC's iSiPP50G platform [85] without compromising the existing functionality of high-speed modulators and photodetectors [83]. And, compact MEMS-based actuators (couplers and phase shifters) controlled by electrostatic comb drives have been demonstrated [10, 86, 87]. Also, a wafer-level sealing technique to protect MEMS devices from outside effects has been developed [88]. The reason of such protection is that suspension of the MEMS devices in air makes the exposed waveguides fragile.

Additionally, Tyndall developed an area-based packaging/assembly approach to interface the MEMS actuators with their drivers using more than 3000 electrical

[3] IMEC, EPFL, KTH, Tyndall, VLC Photonics, and Commscope.

connections [23]. Since the monolithic integration[4] is very expensive [55] and wire bonding of thousands bondpads on the chip edge is not practical, we implemented a high-density ceramic interposer to interface the MEMS actuators to electrical connections to a printed circuit board [83]. High density interposers integration and their corresponding electronic boards enables chip-wide tuning of individual circuit elements. This makes it possible to compensate for variability and parasitics in large-scale circuits, and therefore to increase circuit complexity. And, finally, a software framework [77] for programmable photonics, enabling the design, simulation, visualization, and configuration of FP-PIC circuits has been developed, which also can be extended to other programmable circuits. This framework was one of my main contributions to the project.

To validate the silicon photonic MEMS technology and highlight its multi-functionality (low-power switching, reconfigurability, and redundancy), 3 demonstrators were considered to be implemented:

- A 16×16 optical switch matrix that can be used in communication networks.
- A 16-channel optical beam forming circuit with application in free-space communication, 3D scanning, or LiDAR.
- A 4-channel programmable RF filter that can be used as an arbitrary waveform generator, or a bandpass filter in microwave photonics applications such as radio astronomy or radio-over-fibre (RoF).

These demonstrators were intended to operate in two ways: as a dedicated photonic circuit (or ASPICs), and as a programming scheme in a generic FP-PIC. In fact, apart from the three ASPIC demonstrators, we also implemented a fourth demonstrator: an FP-PIC. the intention was that for the three demonstration cases we could compare the ASPIC with the performance of a programmed FP-PIC.

The circuits were designed in multiple fabrication runs. But during the packaging flow we experienced that the majority of the MEMS devices had mechanically broken down (the movable waveguides had collapsed and stuck to the substrate). Hence, we were not able to demonstrate a working large scale MEMS-based FP-PICS at the end of the project. However, achievements of the project such as modular driver electronics, packaging routines, and software framework can be used as a base for other projects. For example, in the follow up project PHORMIC,

[4] In monolithic integration, all necessary components of the interposer (both electrical and optical) are fabricated directly on the same substrate, typically using silicon photonics technology. This is achieved through advanced CMOS-compatible processes that allow the fabrication of both photonic and electronic components on a single chip.

the project consortium is trying to resolve the issues we faced during MORPHIC, and is working toward the implementation of various advanced and complex circuits by including optical amplifiers.

1.5 IMEC's Silicon Photonic Platform

For this work, we have used IMEC's iSiPP50G [85] as our technology platform, but extended with the new functionalities (i.e. MEMS) developed in the MORPHIC project. Its PDK includes an extensive device library, offering optimized passive and active functions such as waveguides, splitters, and grating couplers for both the C-band and O-band. The IMEC iSiPP50G platform supports submicron silicon waveguides with losses of less than 2dB/cm (0.5dB/cm for rib waveguides), 50Gbps carrier depletion modulators, electro-absorption modulators, and germanium photodetectors. The wafers are finished with a back-end-of-line (BEOL) dielectric stack that includes two copper metallization layers and aluminum bond pads. Although the platform does not currently support integrated light sources, similar to most existing silicon photonics technology platforms, light can be coupled from an external source. Figure 1.9 shows the cross section diagram of the iSiPP50G platform and its extensions to implement actuators based on heaters, liquid crystals, and MEMS.

1.6 Silicon Photonic MEMS Actuators

The two key optical functions that can be implemented by MEMS are power splitting (by couplers) and phase shifting (by phase shifters), and, as discussed before, they are essential to make optical gates for programmable circuits. Project partners at Royal Institute of Technology[5] (KTH) and École Polytechnique Fédérale de Lausanne (EPFL) performed the simulation and design of the MORPHIC phase shifters and couplers. Detailed discussions and analysis can be found in their published works [10, 23, 86, 87, 89]. Here, we briefly summarize the MEMS couplers and phase shifters actuation mechanism used in MORPHIC and present examples of devices that I used in my circuit designs for programmable photonics.

[5] In Swedish, "Kungliga Tekniska högskolan".

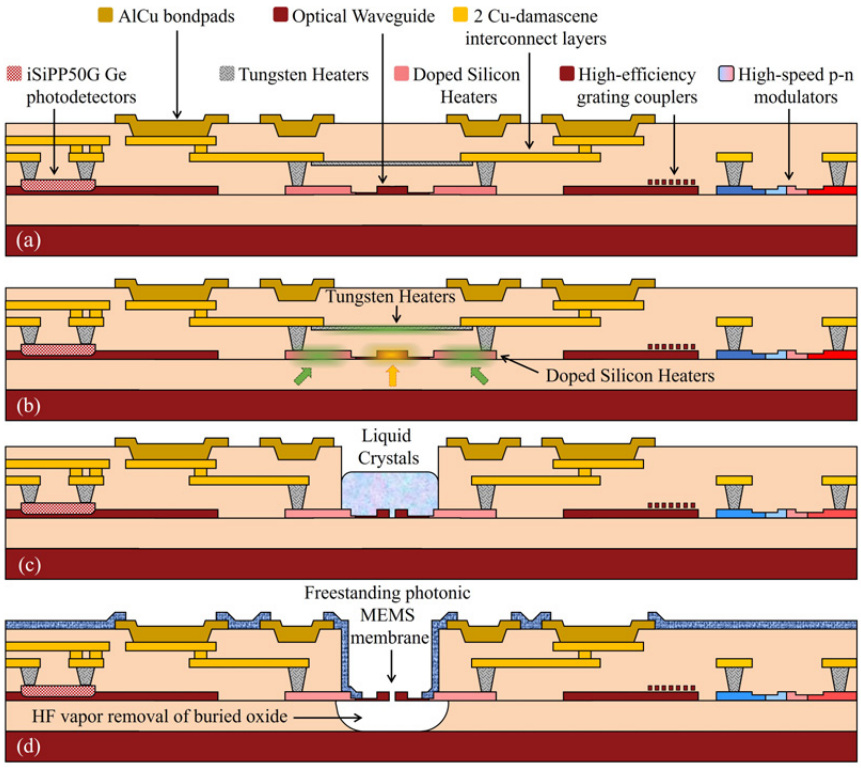


Figure 1.9: a) Cross section diagram of the iSiPP50G platform [5] (heaters are highlighted in (b)) with the extensions for c) liquid crystals, and d) photonic MEMS.

MEMS-tuning approaches rely on the displacement of a structural component generated by an actuator. This process involves the transduction or conversion of energy from one form (electrical) to mechanical energy in the form of motion. MEMS actuators can produce displacements ranging from tens of nanometers to several micrometers. These displacements can be achieved through various methods, such as electrothermal, piezoelectric, and electrostatic transduction. Other methods include magnetic, pneumatic, and shape-memory alloys; however, their integration with PICs is more complex. In the MORPHIC project, we use electrostatic actuation, the most commonly employed displacement mechanism in MEMS. This method relies on the attractive electrostatic force between two oppositely charged bodies. Since charge flow is required only to establish the potential difference and not to maintain it, the power consumption of such devices is nominally zero for steady-state/DC operation. Figure 1.10 shows two simple configurations for out-of plane (fig. 1.10a)

and in-plane (fig. 1.10b) displacement based electrostatic actuation.

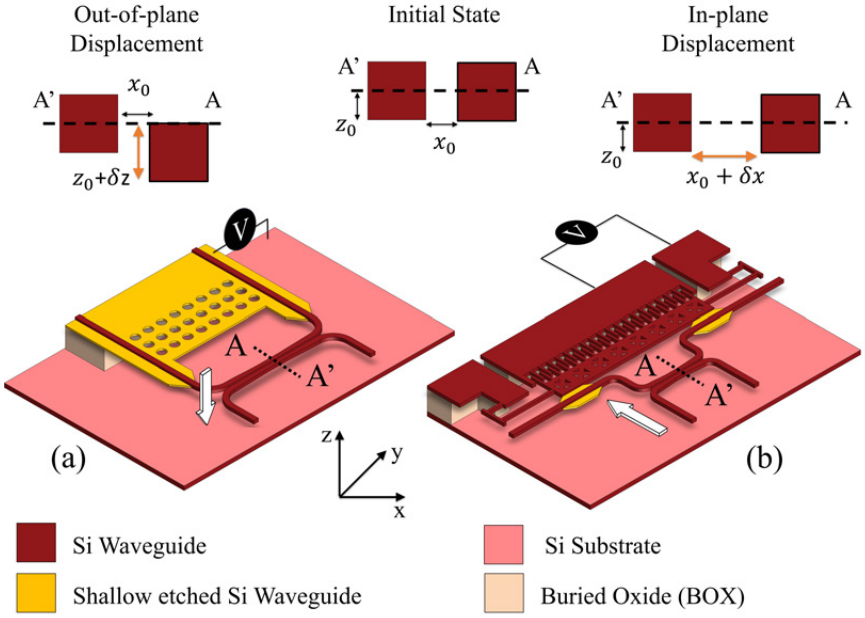


Figure 1.10: Illustration of two electrostatic MEMS actuators: a) one producing out-of-plane displacement. b) in-plane displacement. White arrows show the direction of the displacement when the voltage is applied.

The first configuration consists of a fixed-free (i.e., cantilever) configuration for the suspension and a parallel-plate arrangement for the capacitor. The voltage difference between the suspended plate and the substrate generates an electrostatic force which pulls the plate down. And, the balance between the electrostatic force and the restoring mechanical spring force of the suspension determines the displacement. Within the context of mechanical tuning for PICs, by attaching a waveguide or slab material to the suspended electrode, it is possible to vary the vertical distance (from the line of symmetry between the aligned, i.e., non-actuated, waveguide centers), thereby changing the coupling and phase behavior of the active mode. Figure 1.10b shows a different topology of an electrostatic device, the so-called “comb-drive” actuator. Named for its interdigitated fingers that resemble the teeth/fingers of a comb, this device also uses an attractive electrostatic force to generate displacements. In this case, a voltage difference is placed between the free, suspended and fixed, suspended electrodes so that the resulting electric field pulls the former towards the latter. The restoring force from the suspension exhibits the same linear relationship with spring constant and displacement, but the lateral electrostatic force is slightly

different.

Figure 1.11 summarizes various waveguide arrangements in the MEMS actuators, where evanescence fields of optical modes are illustrated by red auras. As seen, couplers consist of two waveguide cores and phase shifters consist of a waveguide core and a loading structure (a narrow rim of high-index material). These actuators can be fabricated on the single-layer [90] or double-layer SOI wafers [91]. There are two types of displacements; movable parts can move either *in-plane* or *out-of-plane* (vertically) based on the implemented electrostatic actuation mechanism. To induce a phase shift, we can change the effective refractive index of the guided mode by moving the loading structure closer to or further from the core waveguide [89]. In order to avoid power transfer from the waveguide to the loading structure, the waveguide core and loading structure should have significantly different widths. Also, to prevent the loading structure from inducing leakage, absorption, or scattering beyond what is already present in the waveguide itself, the transition geometry should be properly engineered.

Similarly, a tunable 2×2 coupler can be implemented by two parallel waveguide cores, where one of the cores can be displaced to change the gap between two waveguides [92]. For efficient power transfer, the waveguide cores should be symmetric in cross-section to have an identical effective refractive index. Otherwise, the tuning range of the directional coupler is limited [93]. In both couplers and phase shifters, in-plane and out-of-plane movements are enabled by the applied voltage which can induce an attractive force between the fixed and movable elements. In vertical movement, applying a voltage over different layers or the silicon substrate bends a cantilever structure upward or downward [90]. And, for in-plane movement, the suspended beam can be shifted in plane through comb-drive actuators [89,94,95]. We should also mention that, in addition to changing the separation gap between waveguides, it is also possible to vary the amount of overlap (coupling length) between waveguides and consequently tune the coupling or phase shift.

Implementation of photonic MEMS actuators requires both electrostatic actuator design and optical design of the effective index tuning region. Since the evanescent tail of a waveguide dictates the range of gaps that can be used for tuning, target displacement range from transducer region should be set first. Then, an actuator providing desired displacement within a reasonable voltage range should be designed. For directional couplers, a slightly larger travel range should be considered to improve the extinction ratio as much as possible. It is worth to mention that important performance figures-of-merit (FOM) for MEMS-based couplers, switches, and phase shifters include response time, actuation voltage, device

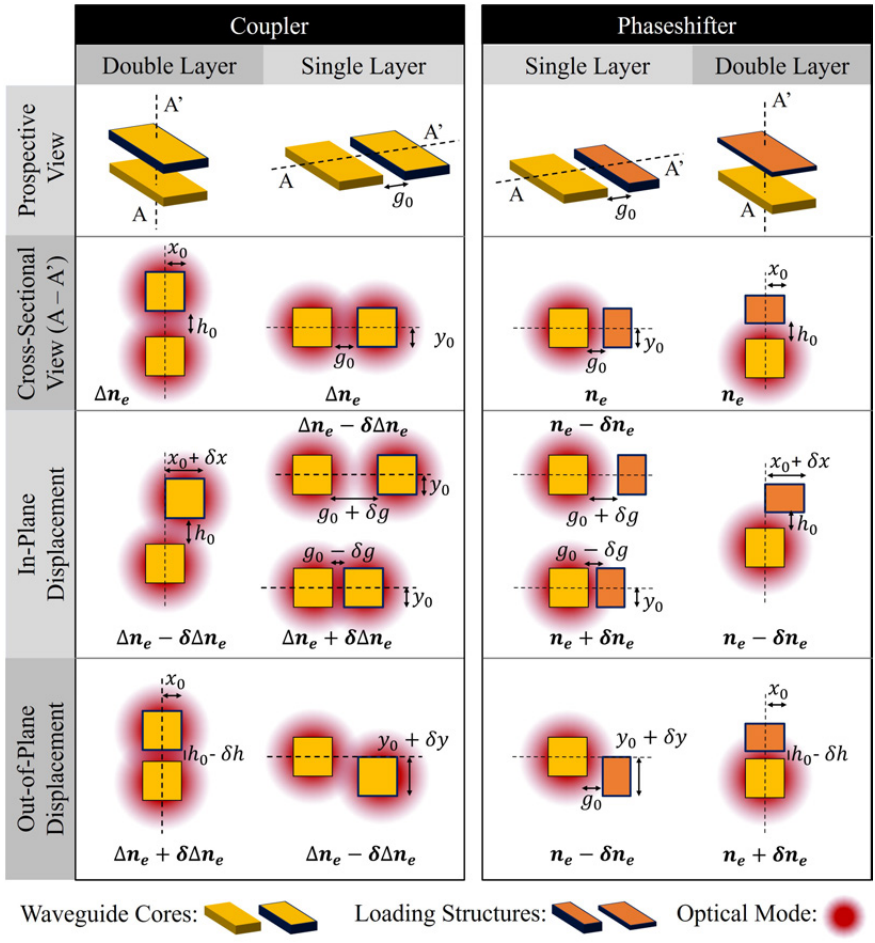


Figure 1.11: summary of the waveguide arrangements in the MEMS actuators. The evanescent fields of optical modes are illustrated by red auras. The structures are regenerated based on [6, 7].

footprint, power consumption, and insersion loss (IL).

1.7 Contribution of this work

The work in this PhD is done as a part of the MORPHIC project with the objective of demonstrating large-scale programmable photonic integrated circuit enabled by integration of MEMS with IMEC's iSiPP50G silicon photonics platform. We had three wafer fabrication run opportunities referred as MORPHIC RUN1, RUN2, and

RUN3, where my designs were fabricated on RUN2 and RUN3. In this project, I was responsible to communicate with both component designers and the packaging group to:

- Design programmable photonic integrated circuits, interposers, and PCB interconnects.
- Test and characterize fabricated photonic chips and electronic control boards.
- Provide feedback for redesign/update of MEMS actuators and electronic boards.
- Develop a framework for design and control of FP-PICs (e.g. Visualization of the configured mesh with corresponding light routing, auto generation of the mesh circuit model, interfacing with low-level programming scripts developed by Tyndall).

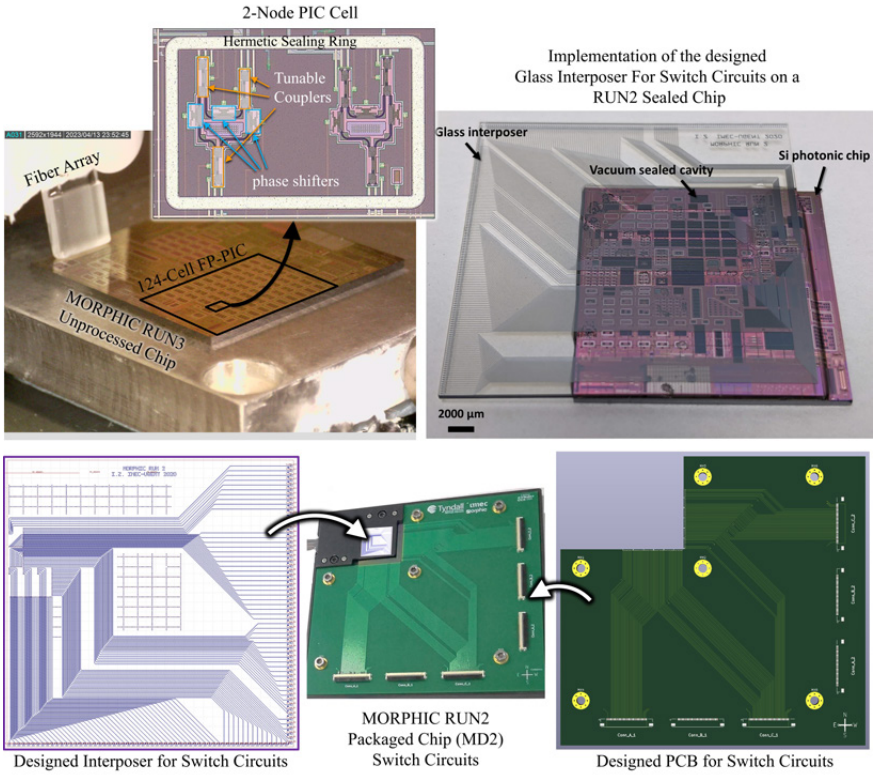


Figure 1.12: Examples of my contribution to the MORPHIC project.

Using the MEMS-based couplers and phase shifters developed by EPFL and KTH, various programmable PICs were designed to demonstrate the functionality and performance of our technology platform. In one of our ambitious designs, I utilized over 700 actuators and 200 photodetectors to create a 9×14 non-uniform hexagonal mesh (Fig. 1.12). For comparison, a 7-cell heater-based circuit was also designed to evaluate the performance of MEMS and thermal actuators.

Since waveguide meshes are central to the optical core of FP-PICs, extensive mesh analysis was conducted to determine suitable architectures for the PICs, taking packaging constraints into account. Various interposers and PCB interconnects for the mini demonstrators were designed (Fig. 1.12), and corresponding design cycles and routines were established. For full demonstrators [6], multi-layer ceramic interposers were used, requiring cross-checking of thousands of electrical routes and connections. A super-netlist containing all design information and optical/electrical connections was created in collaboration with the packaging group in Tyndall.

Several electronic boards (EIC boards, designed by Tyndall) were tested and characterized, and updates were applied to new versions based on our feedback. To facilitate the design and measurement of FP-PICs, I developed a Python-based framework, called Borna. I incorporated IPKISS libraries from Luceda Photonics and low-level programming tools provided by Tyndall, as well as contributions by my colleagues in the Photonics Research Group. Additionally, numerous photonic chips at different stages of fabrication (unprocessed, etched, sealed) were characterized.

Details of my contributions in MORPHIC project are summarized in appendix B.

1.8 Thesis Outline

This thesis consists of 7 chapters. In the introduction chapter we have drawn a overall picture of field programmable photonic integrated circuits (FP-PIC) and MEMS actuators. **Chapter 2** addresses parasitics and digitization effects in hexagonal meshes and MZI-based couplers. **Chapter 3** covers mesh analysis and my circuit designs implemented for fabrication and characterization. In **Chapter 4**, I elaborate on the packaging and electronic control for our programmable circuits, detailing the project's accomplishments and my contributions. **Chapter** 5 discusses the Borna framework, a software interface for controlling our chips. **Chapter 6**

[6] The concept of Full and Mini demonstrators are elaborated in Chapter 4

details the measurement results. And, finally, **Chapter 7** offers conclusions and future perspectives of the project.

1.9 Publications

This thesis has led to the following list of publications in conferences and international peer-reviewed journals.

1.9.1 Publications in international journals

1. N. Quack, A.Y. Takabayashi, H. Sattari, P. Edinger, G. Jo, S.J. Bleiker, C. Errando-Herranz, K.B. Gylfason, F. Niklaus, U. Khan, P. Verheyen, A. Kumar Mallik, J.S. Lee, M. Jezzini, **I. Zand**, P. Morrissey, C. Antony, P. O'Brien, W. Bogaerts, *Integrated Silicon Photonic MEMS , MicroSystems & Nanoengineering*, 9, p.27 (2023).

2. G. Jo, P. Edinger, S. Bleiker, X. Wang, A.Y. Takabayashi, H. Sattari, N. Quack, M. Jezzini, P. Verheyen, **I. Zand**, U. Khan, W. Bogaerts, G. Stemme, K. B. Gylfason, N. Niklaus, *Wafer-level Hermetically Sealed Silicon Photonic MEMS, Photonics Research*, 10(2), p.14 (2022).

3. **I. Zand**, W. Bogaerts, *Effects of Coupling and Phase Imperfections in Programmable Photonic Hexagonal Waveguide Meshes*, Photonics Research, 8(2), p.211-218 (2020).

1.9.2 Publications in international conferences

1. W. Bogaerts, Y. Zhang, H. Deng, L. Van Iseghem, Y. Liu, A. Barzanji, **I. Zand**, A. Ribeiro, U. Khan, K.P. Nagarjun, *General-Purpose Programmable Photonic Circuits*, (invited) publication in IEEE Photonics Society Summer Topicals, Barbados, (to be published).

2. Y. Zhang, X. Chen, L. Van Iseghem, **I. Zand**, H. Salmanian, W. Bogaerts, *A compact programmable silicon photonic circuit*, IEEE Silicon Photonics Conference (2024).

3. W. Bogaerts, H. Deng, X. Chen, L. Van Iseghem, **I. Zand**, Y. Zhang, Y. Liu, A. Al Haffar, P. Edinger, G. Jo, A.Y. Takabayashi, C. Antony, A. Mallik Kumar, J. Zhang, E. Soltanian, P. Verheyen, J. Beeckman, G. Roelkens,

N. Quack, F. Niklaus, K.B. Gylfason, U. Khan, *Building general-purpose programmable photonic chips: opportunities and challenges*, SPIE Photonics West - OPTO (invited), 12890, United States, p.12890-55 (2024).

4. W. Bogaerts, K. Nagarjun, L. Van Iseghem, X. Chen, H. Deng, **I. Zand**, Y. Zhang, Y. Liu, A.Y. Takabayashi, H. Sattari, N. Quack, P. Edinger, G. Jo, S.J. Bleiker, K.B. Gylfason, F. Niklaus, A. Kumar Mallik, M. Jezzini, C. Antony, G. Talli, P. Verheyen, J. Beeckman, U. Khan, *Scaling Programmable Silicon Photonic Circuits*, SPIE Photonics West OPTO (invited), 12426, United States, p.12426-1 (2023).

5. U. Khan, **I. Zand**, L. Van Iseghem, P. Edinger, G. Jo, S. J. Bleiker, A. Y. Takabayashi, C. Antony, M. Jezzini, G. Talli, H. Sattari, J. Lee, A. K. Mallik, P. Verheyen, C. Lerma Arce, M. Garcia, T. Jonuzi, E. Picavet, K. Nagarjun, J. Watte, N. Quack, F. Niklaus, K. De Buysser, J. Beekman, W. Bogaerts, *Low power actuators for programmable photonic processors*, SPIE Photonics West OPTO (invited), 12438, United States, p.124380K (2023).

6. W. Bogaerts, A.Y. Takabayashi, P. Edinger, G. Jo, A.K. Mallik, C. Antony, **I. Zand**, T. Jonuzi, X. Chen, H. Sattari, J.S. Lee, M.A. Jezzini, G. Talli, C. Lerma Arce, S. Kumar, P. Verheyen, N. Quack, K.B. Gylfason, F. Niklaus, U. Khan, *Programmable Photonic Circuits Powered by Silicon Photonic MEMS Technology*, Advanced Photonics Congress - NETWORKS, Netherlands, p.NeM2C.3 (2022).

7. **I. Zand**, C. Antony, X. Chen, W. Bogaerts, *Software Framework Architecture for Programmable Photonic Chips*, IEEE Photonics Society Summer Topicals, Mexico, p.WB1.3 (2022).

8. W. Bogaerts, X. Chen, H. Deng, L. Van Iseghem, M. Wang, **I. Zand**, Y. Zhang, Y. Liu, K. Nagarjun, U. Khan, *Programmable Silicon Photonic Circuits*, OptoElectronics and Communication Conference / International Conference on Photonics in Switching and Computing (OECC / PSC) (invited), Japan, p.TuE4-1 (2022).

9. U. Khan, **I. Zand**, P. Edinger, G. Jo, S. Bleiker, A. Y. Takabayashi, C. Antony, J. Lee, A. Kumar, P. Verheyen, C. Lerma Arce, T. Jonuzi, J. Watte, N. Quack, F. Niklaus, K. Gylfason, W. Bogaerts, *Large scale programmable photonic circuits using silicon photonic MEMS*, Conference on Lasers and Electro-Optics (invited), United States, p.AM2C.5 (2022).

10. W. Bogaerts, L. Van Iseghem, X. Chen, **I. Zand**, H. Deng, M. Wang, K.P. Nagarjun, U. Khan, *Technologies for large-scale programmable photonic circuits*, Progress in Electromagnetic Research (PIERS) (invited), China (2022).

11. U. Khan, **I. Zand**, P. Edinger, G. Jo, S. Bleiker, A. Y. Takabayashi, C. Antony, M. Jezzini, G. Talli, H. Sattari, J. Lee, A. Kumar, P. Verheyen, S. Kumar, C. Lerma Arce, M. Garcia, T. Jonuzi, J. Watte, N. Quack, F. Niklaus, K. Gylfason, W. Bogaerts, *MORPHIC: MEMS enhanced silicon photonics for programmable circuits*, SPIE Photonics Europe 2022, Proc. SPIE (invited), 121480, France, p.121480H (2022).

12. W. Bogaerts, A.Y. Takabayashi, P. Edinger, G. Jo, **I. Zand**, P. Verheyen, M. Jezzini, H. Sattari, G. Talli, C. Antony, M. Saei, C. Lerma-Arce, J. Lee, S. Kumar, M. Garcia, T. Jonuzi, K.B. Gylfason, N. Quack, F. Niklaus, U. Khan, *Programmable Silicon Photonic Circuits powered by MEMS*, SPIE Photonics West OPTO (invited), 12005, United States, p.12005-21 (2022).

13. W. Bogaerts, L. Van Iseghem, X. Chen, **I. Zand**, H. Deng, M. Wang, Y. Zhang, Y. Liu, K.P. Nagarjun, U. Khan, *Programmable Silicon Photonic Circuits*, International Symposium on Photonic and Electronic Convergence (ISPEC 2021) (invited), (2021).

14. N. Quack, A.Y. Takabayashi, H. Sattari, P. Edinger, K.B. Gylfason, G. Jo, F. Niklaus, P. Verheyen, M. Jezzini, U. Khan, **I. Zand**, W. Bogaerts, *Scalable Nano-Opto-Electromechanical Systems in Silicon Photonics*, IEEE Photonics Conference (invited) (2021).

15. W. Bogaerts, L. Van Iseghem, M. Wang, H. Deng, X. Chen, **I. Zand**, K.P. Nagarjun, U. Khan, *Programmable Photonic Circuits*, SPIE Photonex (invited), United Kingdom (2021).

16. W. Bogaerts, L. Van Iseghem, M. Wang, H. Deng, **I. Zand**, X. Chen, K.P. Nagarjun, U. Khan, Programmable Silicon Photonics, International Conference on Solid State Devices and Materials (SSDM) (invited), p.E-201 (2021).

17. W. Bogaerts, A.Y. Takabayashi, P. Edinger, **I. Zand**, G. Jo, H. Sattari, P. Verheyen, M.A. Jezzini, C. Antony, G. Talli, M. Saei, S. Kumar, C. Lerma Arce, M. Garcia Porcel, N. Quack, K.B. Gylfason, F. Niklaus, U. Khan, *Programmable Photonic Circuits using Silicon Photonic MEMS*, Advanced Photonics Congress (invited), p.IM2A.1 (2021).

18. W. Bogaerts, M. Wang, X. Chen, H. Deng, **I. Zand**, L. Van Iseghem, K.P. Nagarjun, U. Khan, *General-Purpose Programmable Photonic Chips*, IEEE Photonics Society Summer Topicals (invited), p.WB2.1 (2021).

19. W. Bogaerts, X. Chen, **I. Zand**, M. Wang, H. Deng, L. Van Iseghem, A. Rahim, U. Khan, *Tutorial: Programmable Integrated Photonics*, European Conference on Optical Communication (ECOC) (invited), p.paper Tu4B-1 (3 pages) (2020).

20. W. Bogaerts, X. Chen, M. Wang, **I. Zand**, H. Deng, L. Van Iseghem, A. Ribeiro, A. Diaz Tormo , U. Khan, *Programmable Silicon Photonic Integrated Circuits*, IEEE Photonics Conference (IPC) (invited), Canada, p.ThD2.1 (2020).

21. W. Bogaerts, A. Rahim, X. Chen, L. Van Iseghem, H. Deng, M. Wang, **I. Zand**, A. Ribeiro, A. Diaz Tormo , U. Khan, *Programmable Photonic Circuits: a flexible way of manipulating light on chips*, Frontiers in Optics (invited), United States (2020).

22. W. Bogaerts, P. Edinger, A.Y. Takabayashi, **I. Zand**, X. Chen, H. Sattari, P. Verheyen, M. Jezzini, G. Talli, S. Kumar, M. Garcia-Porcel, A. Ribeiro, G. Jo, N. Quack, K. Gylfason, F. Niklaus, U. Khan, *Building Large-Scale Programmable Photonic Circuits Using Silicon Photonics MEMS*, OSA Advanced Photonics Congress - Photonics in Switching and Computing (invited), Canada, p.PsTh1F.1 (2020).

23. **I. Zand**, X. Chen, W. Bogaerts, *Application-specific Scaling in Programmable Photonic Circuits*, European Conference on Integrated Optics, France, p.10.3 (2020).

24. W. Bogaerts, A. Ribeiro, M. Wang, L. Van Iseghem, H. Deng, **I. Zand**, X. Chen, U. Khan, *Technologies for large-scale programmable photonic circuits*, AOS Australian Conference on Optical Fibre Technology (ACOFT) and Australian Conference on Optics, Lasers, and Spectroscopy (ACOLS) 2019 (invited), p.11200-55 (2019).

25. **I. Zand**, B. Abasahl, W. Bogaerts, *Intensity Spread Analysis of Programmable Photonic Circuits with Parasitics*, IEEE Photonics Society Summer Topicals, United States, p.paper TuE2.2 doi:10.1109/PHOSST.2019.8794953 (2019).

26. **I. Zand**, W. Bogaerts, Discretization Effects of Digital Control of Thermally Tunable 2x2 MZI Couplers, IEEE Photonics Society Summer Topicals, United States, p.paper TuE2.2 (2 pages) (2019).

27. U. Khan, B. Abasahl, **I. Zand**, N. Quack, K. Gylfason, M. Jezzini, H. Y. Hwang, M. A. G. Porcel, C. Lerma Arce, S. Kumar, W. Bogaerts, *Generic Platform for Silicon Photonics based on MEMS Reconfigurable Photonic Integrated Circuits*, Photonics & Electromagnetics Research Symposium (PIERS) 2019 (invited), Italy, p.2271 (2019).

28. B. Abasahl, **I. Zand**, C. Lerma Arce, S. Kumar, N. Quack, M. A. Jezzini, H. Y. Hwang, K. Gylfason, M. A. G. Porcel, W. Bogaerts, Towards *Low-Power Reconfigurable Photonic ICs Based on MEMS Technology*, Australian Institute of Physics Congress (invited), Australia, (2018).

29. **I. Zand**, B. Abasahl, U. Khan, W. Bogaerts, *Controlling parasitics in linear optical processors*, Proceedings of the 23rd Annual Symposium of the IEEE Photonics Benelux Chapter, Belgium, p.152-155 (2018).

2

Parasitics and Discretization Effects in Optical Gates

In this chapter, the effect of imperfections on the transmission and crosstalk in programmable photonic meshes with feedback loops consisting of tunable couplers and phase shifters are studied. The many elements in such meshes can generate a multitude of parasitic paths when the couplers and phase shifters deviate even slightly from their nominal value. Performing Monte Carlo simulations, we show that small stochastic imperfections in the phase and coupling ($< 1.0\%$) can introduce unwanted interferences and resonances and significantly deteriorate the frequency response of the circuit. We also demonstrate that, in the presence of imperfections, the programming strategy of the unused couplers can reduce effects of such parasitics.

In the second part of this chapter, we describe how digital voltage driving of tunable 2×2 MZI-based couplers with thermo-optic and MEMS phase shifters introduces discretization errors which significantly affect programmable photonic circuits. Performing quantitative analysis, we show that proper biasing of couplers and simultaneous driving of arms can improve discretization errors.

This work has been published in one journal paper [81] and three conference papers [78–80].

2.1 Parasitics

The physical implementations of programmable photonic meshes consist of many identical optical building blocks (phase shifters and tunable couplers), and the routing of the light is controlled by electronics and software. Such a chip, therefore, requires electronic drivers for all the optical elements, an assembly scheme for fibers and high-speed connections, and algorithms and software that will allow a designer to implement a useful function. Hence, there are various possible sources of errors, both in design and fabrication, causing imperfect behavior of the optical building blocks, which in turn will introduce parasitic behavior in the mesh. These errors (acting as additional loss, phase errors, and coupling errors in each of the gates) can accumulate and dramatically deteriorate the response of the circuit [30, 96]. In particular, for the meshes with feedback loops, they create a multitude of secondary and tertiary paths for the light, which can cause unwanted interferences and resonances and, thus, affect the frequency response of the circuit [30]. This will result in a wavelength-dependent transmission "ripple" in the desired output ports and crosstalk in the other ports. We can discern three types of parasitics originating from the phase shifters and tunable couplers:

- Nonidealities in the optical transmission where light remains in the intended waveguide paths. This can be an error in the phase shift or an error in the coupling ratio. These, in principle, could be compensated with improved control electronics and algorithms of the tunable building blocks.
- Nonidealities in the components where light is coupled to another waveguide path, where it should never end up during normal operation. The most common example is backscattering in waveguides and backreflection/backcoupling in tunable couplers [97].
- Nonidealities that radiate light from the waveguides altogether. Due to reciprocity, such defects can also capture light radiated elsewhere on the chip, which can introduce secondary light paths.

We should point out that all these nonidealities might have wavelength-dependent behaviour. Hence, cancelling them out around one wavelength might not eliminate them over the full spectrum.

Here, we focus on the first type of nonidealities and study the effect of small stochastic imperfections in the phase and coupling control of a 7-cell hexagonal mesh consisting of 2×2 couplers and phase shifters connected by waveguides. For the simulations, we have extended the photonic circuit simulator *Caphe* (part of the design framework IPKISS) [98] with models for the mesh components and

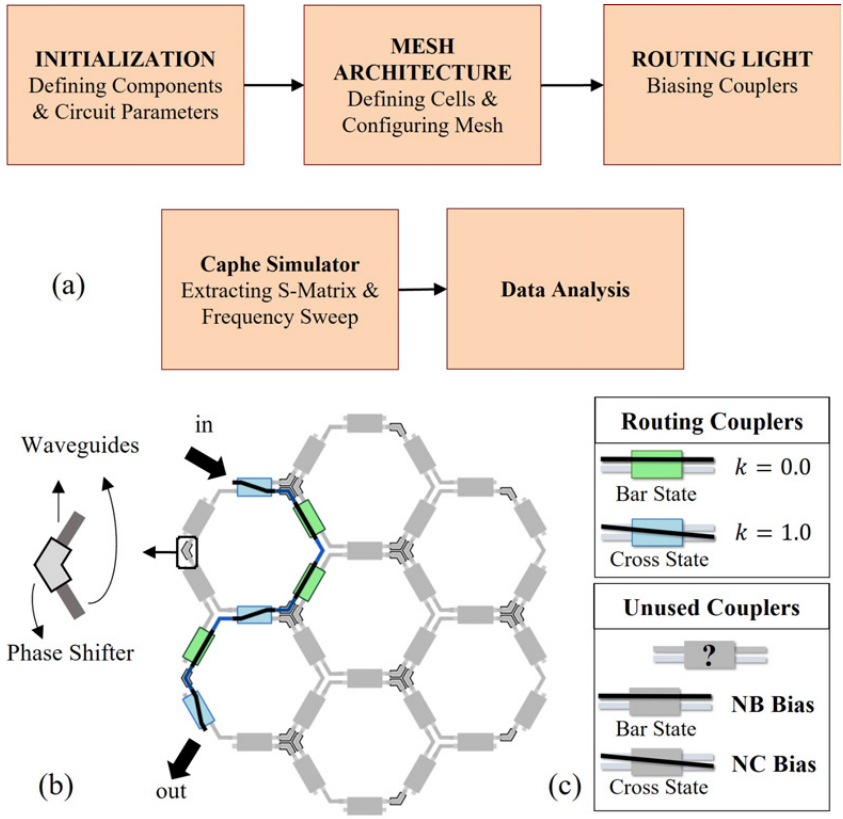


Figure 2.1: Summary of the simulation flow to study effects of parasitics. (b) Schematic representation of the 7-cell hexagonal mesh, where 2 × 2 couplers are connected to the phase shifters (PSs) through silicon waveguides. (c) For each mesh configuration, couplers are categorized to routing couplers (involved in defining light paths) and unused couplers (their state, in principle, does not affect the light paths). Orange and gray colors show cross and bar states of the routing couplers; unused couplers are shown only by the blue. Note that, in normal bar bias (NB bias), unused couplers are programmed in the bar state, while, in normal cross bias (NC bias), they are programmed in the cross state.

performed Monte Carlo simulations to evaluate the effect of imperfections. The summary of the simulation flow is shown in Fig 2.1(a).

2.1.1 Simulation Methodology

As a baseline case for the simulations, we have constructed a uniform mesh based on hexagonal cells, which offers flexible programming, especially because this mesh type allows clockwise/ counterclockwise coupling [99]. Figure 2.1(b) shows

a schematic representation of a hexagonal mesh consisting of seven hexagonal cells. As seen, the arms of the 2×2 couplers (CPs) are connected to the phase shifters (PSs) through silicon waveguides, where optical length of the phase shifters is assumed to be zero and where they only apply the desired additional phase shifts. We also assume that perfect control over the variable property of each component is feasible, and this is over its entire operational range. This limit is 0–100% coupling efficiency for the CPs and 0°–360° phase delay for the PSs; further, all elements can be controlled independently, ignoring at this stage effects of tuning crosstalk (e.g., through thermal crosstalk). The total length of each segment is assumed to be $300\,\mu\mathrm{m}$ with a waveguide bend radius of $10\,\mu\mathrm{m}$.

Our circuit generator is implemented using Python on top of the IPKISS/Caphe framework by Luceda Photonics, which generates hexagonal lattices with arbitrary configurations of the cells. Mesh components are implemented as parametric cells in the IPKISS framework, so they contain a layout, connectivity information (netlist), and a circuit model. This means that both realistic and abstract building blocks of the components can be used for the calculations, and the generated layouts can even be used for fabrication. Also, as the layout feeds back into the circuit simulation, actual waveguide lengths and device parameters are being used. Even though the components are considered identical in the entire mesh, we can set the circuit model parameters for each instance individually to "program" the circuit, using the *variability extensions* for the Caphe simulator developed in our group [100]. In the Caphe circuit simulator, the optical properties such as phase shift (for the PS blocks), coupling coefficient (for the CP blocks), and insertion loss (for both) can be varied. We can also visualize the model parameters for each building block as part of the overall circuit. This can be used to test programmability and also to evaluate sensitivity of the circuit to variations.

In order to evaluate the response of the circuit subject to parasitic imperfections, we consider deviations from the assigned values for the phase delay of the PSs and the coupling coefficient of the CPs. Otherwise, the rest of the properties are assumed to be unperturbed. The errors are described as a normal distribution around the parameters' nominal values with varied standard deviations (hereafter σ) considered. In the Monte Carlo simulations, a population of 100 experiments is defined for which the parameter values are randomly generated.

In a programmable photonic circuit, routing is the most prevalent functionality to be implemented, as it is used to interconnect all other functions. For simple routing, couplers will only be configured in either cross (coupling coefficient $\kappa = 1$) or bar state ($\kappa = 0$). In contrast with ordinary circuits, where routing is done through static waveguides, in a programmable circuit, the routing is done by the tunable couplers. As a result, there is a much higher possibility for light to leak

into adjacent paths and cause undesired interference within the circuit. The couplers, which are used to create the actual route, are referred to as routing couplers (RCPs), and remained couplers in the mesh are the unused couplers (UCPs); UCPs are not involved in the routing, and in ideal circumstances, there should be no light passing through those elements. Further, under perfect operation conditions, their state does not affect the path shape. However, as we will see, their state does affect the function of the circuit in the presence of imperfections. Therefore, we define two different biasing schemes: setting all the UCPs in bar state or cross state. We refer to these biases as "normal bar" (NB) and "normal cross" (NC), respectively (Fig. 2.1c). For the schematic reperesentation of the unused and routing couplers, we have chosen the blue color for the UCPs and gray/orange color for the RCPs when they are in the bar/cross state, respectively.

2.1.2 Single Paths

Figure 2.2(a) shows a path (with $L_{path} = 6L_u$, L_u is the unit length and includes a coupler, a phase shifter, and a bend waveguide) routed through a 7-cell mesh, and Fig. 2.2(b) plots the transmission from the input to the output port for NB bias (red lines) and NC bias (green lines), with random variations in the couplings κ with $\sigma_\kappa = 0.05\%, 0.4\%, 1.0\%$, and variations in the phase delays ϕ with $\sigma_\phi = 17^\circ$. When the mesh is ideally programmed (black dashed lines), we expect that the length of the path will only contribute losses. However, as the graph for the NB bias shows, when random coupling variations increase, we see that levels of transmissions drop further, and significant ripples appear on the output spectrum. In fact, unwanted coupling will introduce additional losses because light is tapped out of the main path. Further, it also introduces parasitic interference paths and even ring resonators. In fact, with couplers in NB state, we can form natural ring resonators, which are of course again coupled together through the imperfect couplers.

Although additional losses caused by unwanted couplings from the main path are inevitable, it is expected that proper programming of the unused couplers (using NC bias) suppresses many parasitic interference paths and prevents excessive ripples of the transmission response. As shown in Fig. 2.2(b), using NC bias can successfully compensate light accumulation in the mesh and suppress the formation of the coupled ring resonators in the mesh. As a result, the ripples of transmission have been suppressed.

To analyze the response of the programmed mesh for the NB and NC biases, we collected the 5%–95% intensity spread in the transmission spectra of Fig.2.2(b) for 100 Monte Carlo simulations with the different random coupling variations of

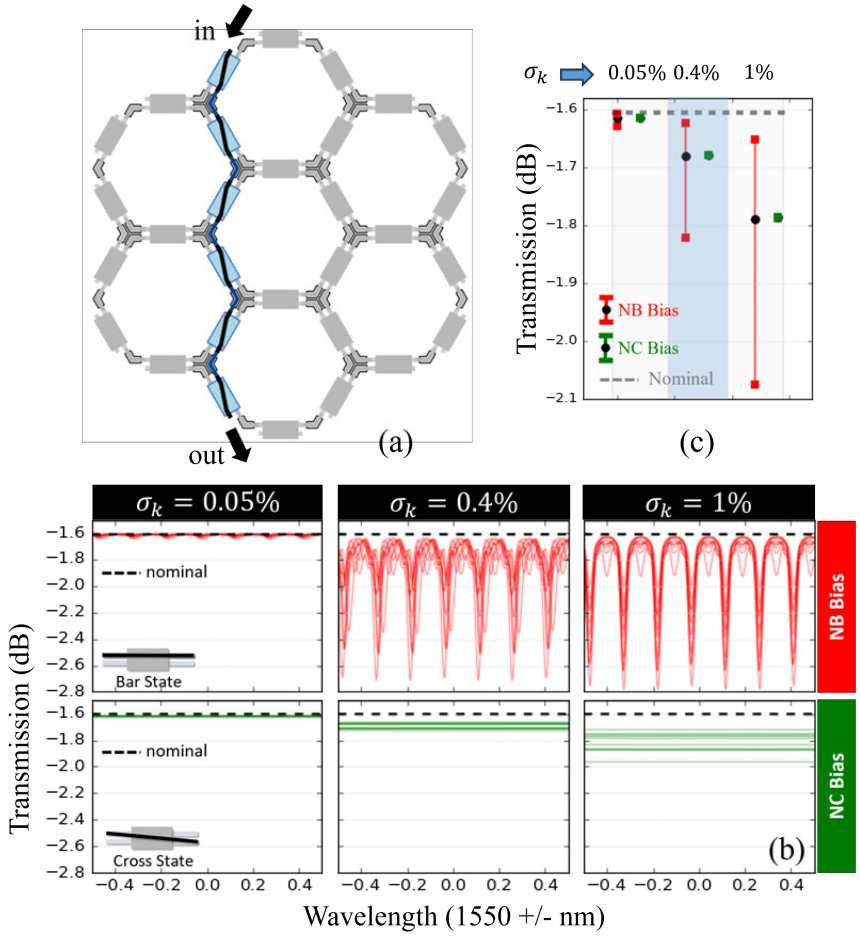


Figure 2.2: (a) Schematic of a routed path (with $L_{path} = 6L_u$) within a 7-cell mesh. (b) Transmission spectra of the mesh for two types of biasing: normal bar (NB), where unused couplers are biased in the bar state (red curves), and normal cross (NC), where unused couplers are biased in the cross state (green curves). The results are plotted for $\sigma_\kappa = 0.05\%, 0.4\%, 1.0\%$ from left to right. (c) Intensity spread analysis of the transmission in the output for random variations of $\sigma_\kappa = 0.05\%, 0.4\%, 1.0\%$. Red and green error bars correspond to the NB and NC biases.

$\sigma_\kappa = 0.05\%, 0.4\%, 1.0\%$, and plotted them as red (NB bias) and green (NC bias) error bars in Fig.2.2(c). The [min, mean, max] points of the error bars are obtained by:

$$min = min\{min[T(\lambda)]_5, ..., min[T(\lambda)]_{95}\} \tag{2.1}$$

$$mean = mean\{mean[T(\lambda)]_5, ..., mean[T(\lambda)]_{95}\} \tag{2.2}$$

$$max = max\{max[T(\lambda)]_5, ..., max[T(\lambda)]_{95}\} \tag{2.3}$$

The subscripts 5 and 95 indicate that we only considered the 5th–95th percentile of the samples, discarding the most extreme values. Comparing error bars clearly shows that NC bias of the unused couplers considerably reduces the intensity spread of the transmission ($>$95%).

We performed this intensity spread analysis on a variety of simple and complex paths for NB and NC biases (Fig.2.3). Similar to Fig.2.2(c), error bars are plotted for three different values of σ_κ and compared with the nominal response of the path (shown by black dashed lines). Each configuration is illustrated either above or below its corresponding intensity spread error bars. As seen, for NB bias, larger σ_κ causes more intensity spread (larger ripples in spectrum) and losses, and complex paths with loops in their configuration (L, M, N) have error bars with a maximum transmission higher than the nominal values; in fact, effects of parasitic shortcuts (shorter paths with lower loss than the main path) are more prominent and can result in constructive interference of the output signal arriving through shortcuts. On the other side, for the NC bias, the effect of parasitic interference is almost eliminated for the simple paths; in addition, for the more complex paths, although this benefit diminishes, we still see a reduction of 50% in the intensity spread compared to NB bias.

It is worth noting that resonances are an artifact that only occurs in recirculating meshes. Forward-only meshes [12, 16] can also suffer from parasitics, but there we are only considering feed-forward interferences. Still, in the case of a single parasitic beam, this can still lead to a 10% intensity fluctuation when the parasitic beam carries only 1% of power. A second advantage of feed-forward meshes is that they are easier to control with simple minimizing/maximizing feedback loops [12], which can help to reduce the parasitics in real time.

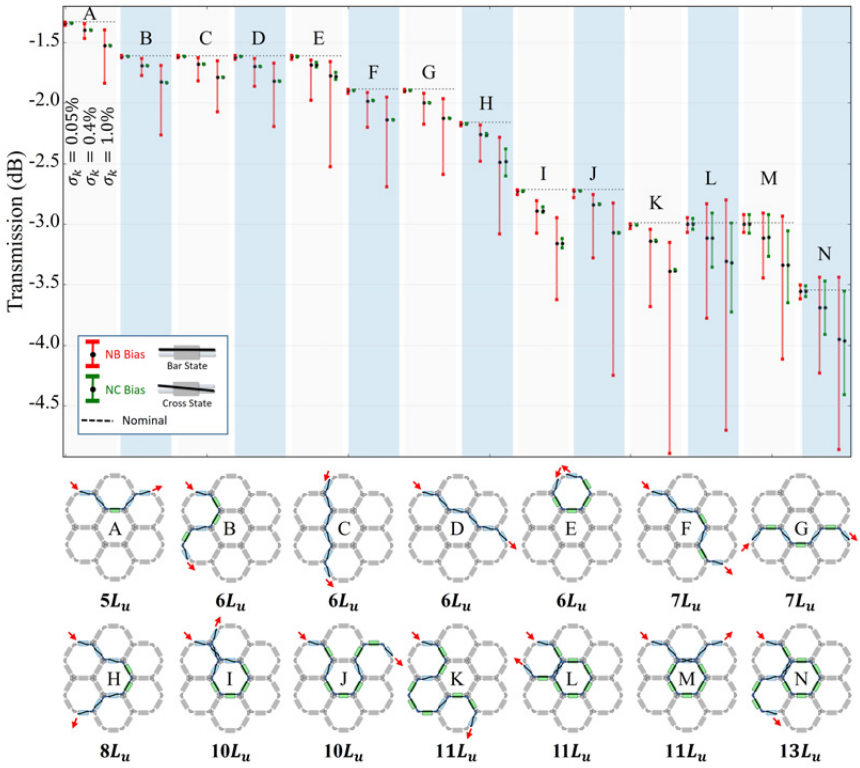


Figure 2.3: Intensity spread analysis of different configurations of the 7-cell mesh to study both simple and complex paths. Blue couplers are in the bar (cross) state for the NB (NC) bias. Random variations of $\sigma_\kappa = 0.05\%, 0.4\%, 1.0\%$ are chosen for the Monte Carlo simulations. Red and green error bars correspond to the NB and NC bias, respectively.

2.1.3 Multipath Routing

The advantage of a programmable circuit is that we can implement multiple functions at the same time, connect multiple inputs to multiple outputs, and even use the tunable couplers as crossings to make intersecting paths. Of course, when there are imperfections in such scenarios, it is important that crosstalk between different paths is kept to a minimum. Figures 2.4(a) and 2.4(b) show the intensity spread of the transmission and crosstalk spectra of a mesh configured for the vertical and horizontal double-paths (A and C) and a multipath composed of them (B).

It is clear from Fig. 2.4 that putting the circuit in NC bias (green bars) eliminates parasitic shortcuts for all the cases, and their transmission intensity spread reaches 0 dB. This means that, similar to Fig. 2.2(b), all ports will have an almost flat

transmission response without ripples. However, increasing σ_κ will increase losses and reduce transmitted power, which is inevitable. In NB bias, the effect of parasitics is noticeable for the double-paths (A and C), while it has been reduced for the C configuration. The reason is that the C configuration has more couplers in the cross state, which breaks loops in the mesh and stops light accumulation inside. For the crosstalk, plotted in Fig. 2.4(b), increasing σ_κ increases its value for all configurations for both NB and NC biases. It is also seen that outputs in A (O_1 and O_2) and C (O_3 and O_4) configurations have higher crosstalk compared with the B configuration.

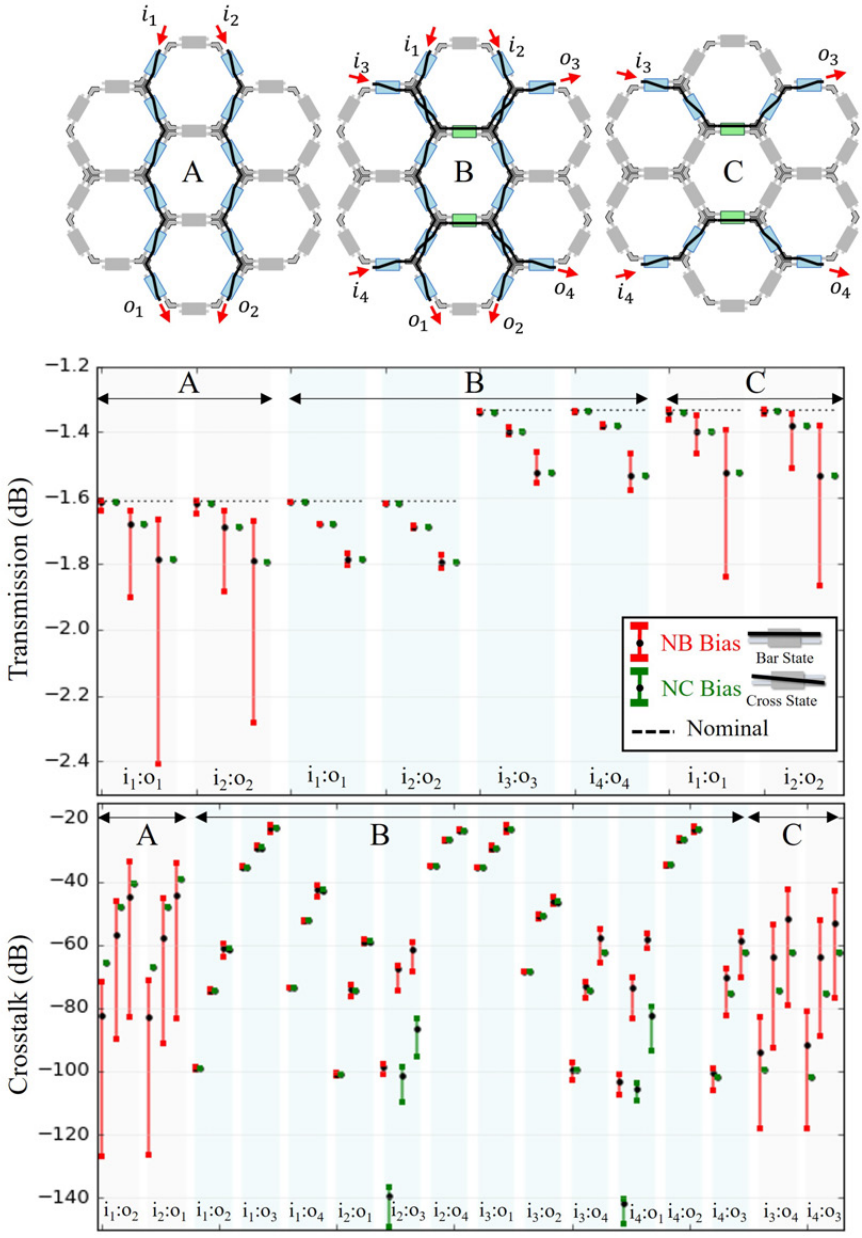


Figure 2.4: Intensity spread analysis of the (a) transmission and (b) crosstalk of double- and multipaths. Similar to the Fig. 2.3(c), error bars are plotted for $\sigma_\kappa = 0.05\%, 0.4\%, 1.0\%$.

2.1.4 Mach–Zehnder Interferometers

Mach–Zehnder Interferometers (MZI) are the basis of all finite impulse- response filters. The performance of a single MZI is a measure of how easily and accurately higher-order filters can be implemented. A simple MZI can also be easily evaluated for imperfections by inspecting the extinction ratio and absolute wavelength registration.

Figure 2.5(a) shows three different configured MZIs (A:$\Delta L = 6L_u$, B: $\Delta L = 4L_u$, C: $\Delta L = 10L_u$). The corresponding transmission responses of these three configurations are plotted in Fig. 2.5(b), where we have again used red and green curves for NB and NC biases. Here, to show the effect of NB and NC biases, 10 cycles of Monte Carlo simulations are used for $\sigma_k = 1\%$ and $\sigma_\phi = 17^\circ$. Also, for simplicity, only one of the outputs (O_1) is shown (the other one, O_2, shows similar behavior). As seen, while the coupling errors mainly cause deterioration in the extinction ratio and weak appearance of other harmonics, the error in the phase shifters introduces a redshift or blueshift in the spectrum. Although phase shifter errors deteriorate the responses for NC and NB biases, NC bias shows better extinction ratios.

In order to quantify performance of the circuit for the selected MZI configurations, a correlation-based analysis has been performed using 100 cycles of Monte Carlo simulations; the results are shown in Fig. 2.5(c). Here, three pairs of coupling and phase variations (σ_κ, σ_ϕ) have been selected: (0.05%, 17°), (1.0%, 17°); and (1.0%, 2°). For this analysis, the nominal response without perturbation is considered as the reference signal (R); for the perturbed responses (S), the correlation with R ($Corr(S,R)^2$) is calculated, where $Corr(S,R)^2 = (\int S.R^*\, d\lambda / \sqrt{\left(\int |S|^2\, d\lambda\right)\left(\int |R|^2\, d\lambda\right)})^2$. The autocorrelation of the reference ($Corr(R,R)^2$) has its maximum at the zero shift ($Corr(R,R)^2|_{\Delta\lambda=0}$), and both correlation functions show small local maxima due to their periodic behavior. The closer the period of the two signals, the more similar these local maxima. The correlation graphs are also normalized with respect to the autocorrelation of S ($Corr(S,S)^2$) and R ($Corr(R,R)^2$). In other words, the normalized correlation function can be read as:

$$\widetilde{Corr}(S,R)^2 = \frac{Corr(S,R)^2}{\sqrt{Corr(S,S)^2|_{\max} \times Corr(R,R)^2|_{\max}}} \tag{2.4}$$

The value $\widetilde{Corr}(S,R)^2|_{\Delta\lambda=0}$ corresponding to the normalized correlation of S with R is an indicative value of the resemblance of the nominal and perturbed responses. At a shift wavelength of $\Delta\lambda_{\text{MAX}}$, the normalized correlation

curve of S has a maximum $\widetilde{Corr}(S,R)^2|_{\text{MAX}}$. This indicates that, on average, S has a $\Delta\lambda_{\text{MAX}}$ shift as compared with R; if such a shift was absent (or unimportant depending on the application), the resemblance value of S and R would be $\widetilde{Corr}(S,R)^2|_{\underline{\text{MAX}}}$. Figure. 2.5(c) shows calculated values of $\widetilde{Corr}(S,R)^2|_{\Delta\lambda=0}$, $\Delta\lambda_{\text{MAX}}$, and $\widetilde{Corr}(S,R)^2|_{\text{MAX}}$ for the configured MZIs and different pairs of (σ_k, σ_ϕ), where we have used error bars to compare NB (red) and NC (green) biases. As seen, phase error, compared with coupling error, is the prominent factor, and, by reducing it to 2°, we can achieve acceptable resemblance with the nominal response. It is also seen that NC bias is not as efficient as it was for basic routing; however, it will probably still improve the response because the MZI will also suffer from the parasitic resonances, but it does not affect the wavelength shift.

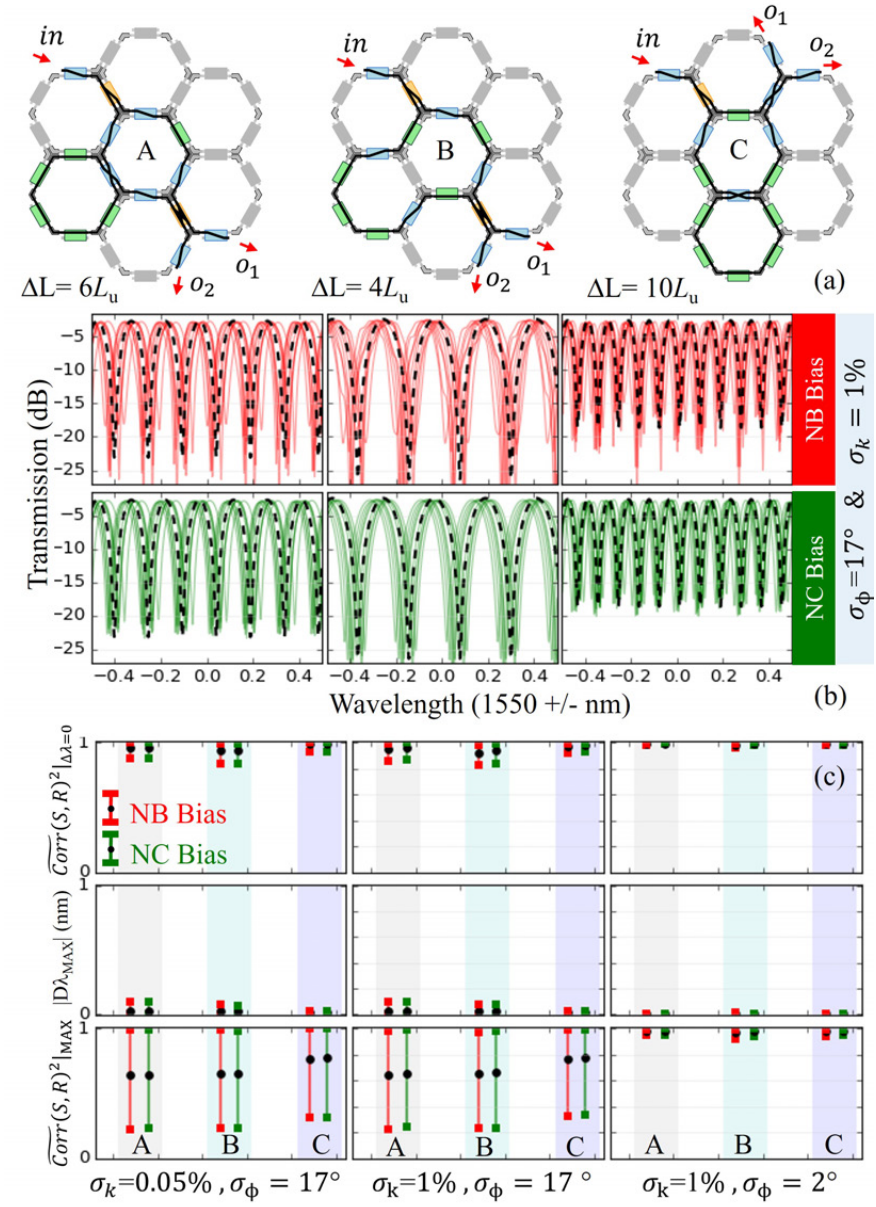


Figure 2.5: (a) Schematic of the three different configured MZIs in the 7-cell hexagonal mesh (A: $\Delta L = 6L_u$, B:$\Delta L = 4L_u$, C: $\Delta L = 10L_u$). b) Transmission response of the MZIs for NB (red) and NC (green) biases, where only 10 cycles of the Monte-Carlo simulations have been plotted for better visibility. c) Correlation-based analysis of the configured MZIs for (σ_k, σ_ϕ) pairs of (0.05%, 17°), (1.0%, 17°), *and* (1.0%, 2°).

2.1.5 Ring Resonators

Ring resonators, which are the building blocks of many silicon photonics filters, are susceptible to peak-splitting due to backreflection [101, 102]. While we did not incorporate backreflection in the parasitic analysis (they are considered as a second type of parasitics), the hexagonal mesh allows for clockwise/counterclockwise coupling through the coupling between two adjacent rings, which are not originally designed to share any signal. This has a similar effect as backreflection or backcoupling in the tunable couplers. Figure 2.6(a) shows three configured ring resonators (A, B, C) in our chosen 7-cell hexagonal mesh. For the A and C configurations, rings are located in the center of the mesh; however, their bus waveguides are routed differently. For the B configuration, the ring is located near the boundary of the mesh, and its bus waveguide is created by a long and complex path. Transmission responses of these configurations for NB (red) and NC (green) biases with $\sigma_\kappa = 1\%$ and $\sigma_\phi = 17^\circ$ are plotted in Fig. 2.6(b), where 10 cycles of Monte Carlo simulations have been used. As expected, the phase errors mainly have caused wavelength shifts for all configurations and the coupling error affects the extinction ratio. However, complexity of the path in configuration B has created a larger extinction ratio.

We also have analyzed the performance of the ring resonators using the same correlation technique used for the MZI analysis. Figure 2.6(c) shows calculated values of $\widetilde{Corr}(S,R)^2|_{\Delta\lambda=0}$, $\Delta\lambda_{\text{MAX}}$, and $\widetilde{Corr}(S,R)^2|_{\text{MAX}}$ for the configured resonators and different pairs of (σ_k, σ_ϕ). Similar to the MZIs, phase errors can considerably change the shape of the circuit response and by reducing σ_ϕ to 2° we can achieve a resemblance of more than 90%. Another observation is that, although using NC bias cannot eliminate the effect of phase errors, selecting the ring close to the boundaries (configuration B) can significantly improve performance of the circuit. Comparing configurations A and C also shows that the routing of the path can also affect the response of the circuit even if the rings are in the same location.

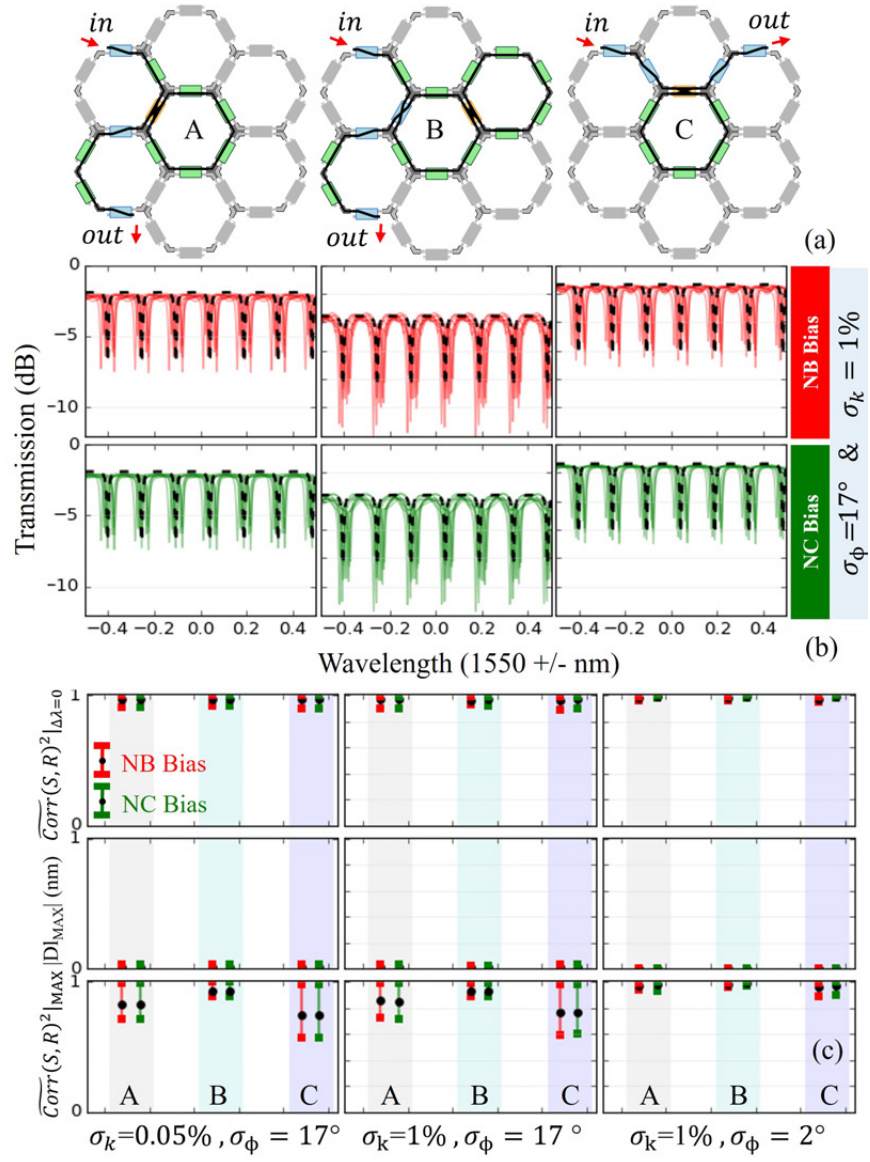


Figure 2.6: (a) Schematic of three different configured ring resonators in the 7-cell hexagonal mesh. (b) Transmission response of the selected configurations for NB (red) and NC (green) biases, where 10 cycles of the Monte Carlo simulations have been used. (c) Correlation-based analysis of the configured ring resonators for (σ_κ, σ_ϕ) pairs of (0.05%, 17°), (1.0%, 17°), and (1.0%, 2°).

2.1.6 Splitters and Power Distribution Networks

Another important configuration in programmable circuits is splitter, which can be used for multicasting or as a distribution network for an optical beam former. Here, we present an intensity spread analysis of a 1×4 (Fig. 2.7) and 1×16 (Fig. 2.8) splitter network. Similar to the previous cases, two biasing schemes (NB and NC biases) are compared, and random variations of $\sigma_\kappa = 0.05\%, 0.4\%, \text{and} 1.0\%$, and $\sigma_\phi = 17^\circ$ are applied to the couplings κ and phase shifts ϕ, respectively.

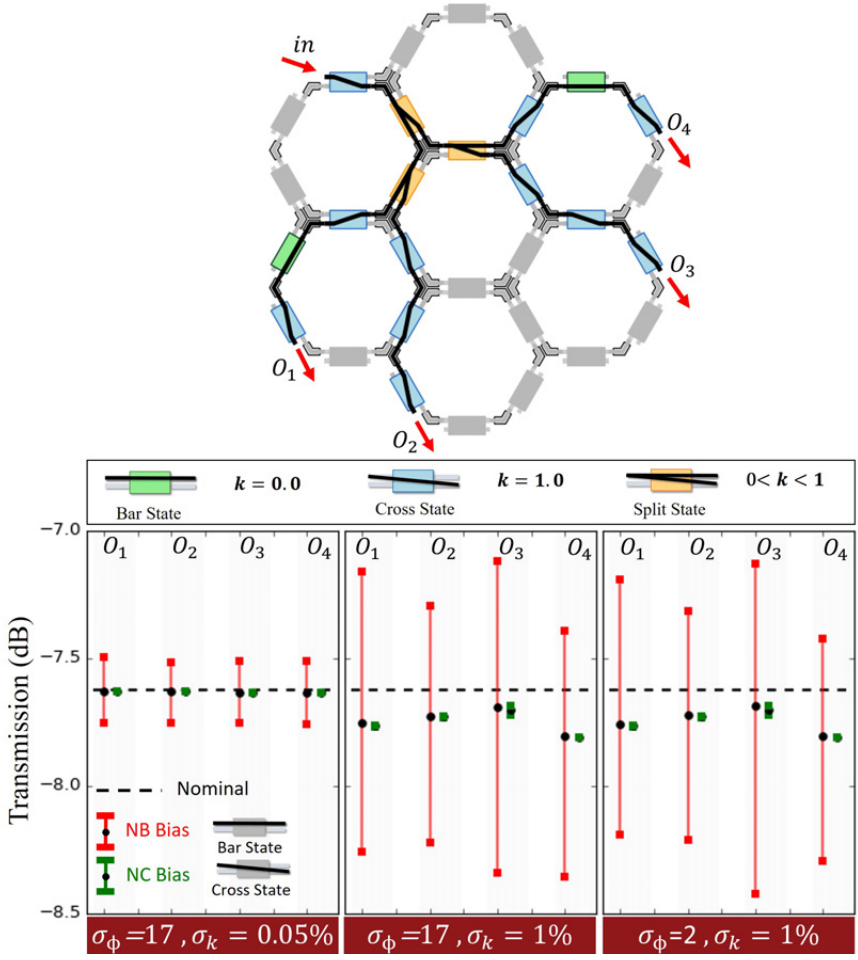


Figure 2.7: Intensity spread analysis of a 1×4 splitter in the 7-cell hexagonal mesh. Red and green bars show NB and NC biases, respectively. Similar to Fig. 2.2(c), error bars are plotted for $\sigma_\kappa = 0.05\%, 0.4\%, \text{and} 1.0\%$. For NC bias, blue couplers are in the cross state ($\kappa = 1$), while they are in the bar state ($\kappa = 0$) for NB bias.

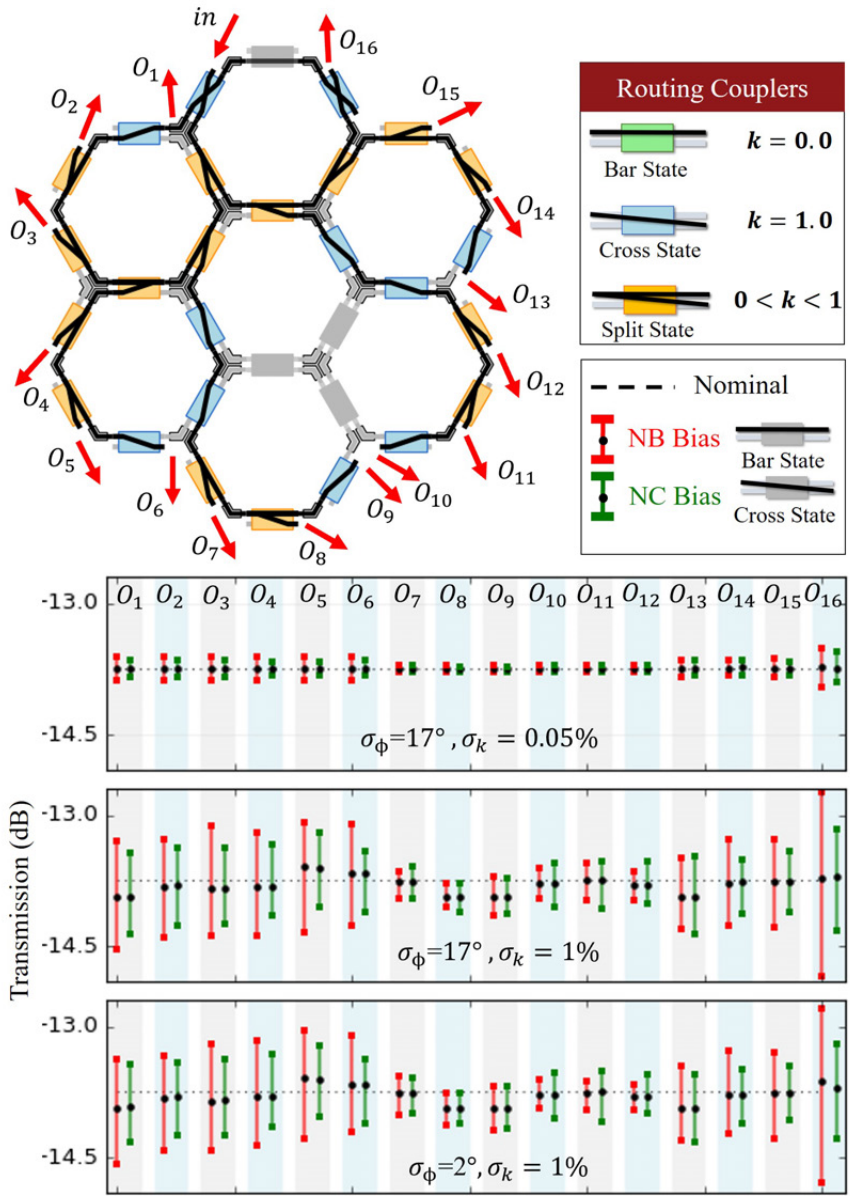


Figure 2.8: Intensity spread analysis of a 1 × 16 splitter in a 7-cell hexagonal mesh. Red and green bars also show NB and NC biases, respectively. Similar to Fig. 2.2(c), error bars are plotted for $\sigma_\kappa = 0.05\%, 0.4\%, and 1.0\%$. For NC bias, blue couplers are in the cross state ($\kappa = 1$), while they are in the bar state ($\kappa = 0$) for the NB bias.

As seen in Fig. 2.7, similar to the simple routing paths, the coupling error is the dominant factor that affects the power transmission. For NB bias, increasing σ_κ from 0.05% to 1.0% results in 0.25 dB to 1.8 dB power variation in the transmitted signals to the outputs. Also, higher transmissions are seen compared with their nominal response, indicating parasitic shortcuts, as discussed for the paths. However, applying NC bias successfully compensates the effect of unwanted interferences caused by parasitic couplers, and intensity spread reduces by more than 95%.

For the 1 × 16 splitter, almost all ports and coupler elements of the mesh are actively used; further, the circuit is operating near its full capacity. Hence, there is small room to compensate the effect of parasitics using unused couplers. This can be observed by comparing green (NC bias) and red (NB bias) error bars in Fig. 2.8. As seen, intensity spread reduction by the NC bias is less than 0.2 dB. Also, similar to the 1 × 4 beam splitter, parasitic shortcuts have increased power level of the transmitted signal. Another observation is lower intensity spread of the O_7-O_{12} ports compared with the others (50% less).

2.1.7 Customized Biasing Schemes

As discussed above, by using NC bias, parasitic effects can be eliminated considerably for the paths and splitters. However, setting all of the unused couplers in the cross state is not the only solution for parasitic elimination. In fact, by proper programming of some of unused couplers, the same or better results can be achieved. To elaborate on this, transmission spread analyses of a single path ($\Delta L = 8L_u$) for the NB bias, NC bias, and three other customized biases are shown in Fig. 2.9, where $\sigma_\kappa = 0.05\%, 0.4\%,$ and1.0%. As seen, by programming only a few unused couplers, much better results can be achieved; for the biasing scheme of E, the transmission spread has been reduced by 95% for $\sigma_\kappa = 0.1$. The importance of such optimization is in consuming lower energy to eliminate parasitic effects. Hence, for future studies, optimization of algorithms and strategies can be studied in more depth to find the optimum solutions for proper biasing of desired configurations.

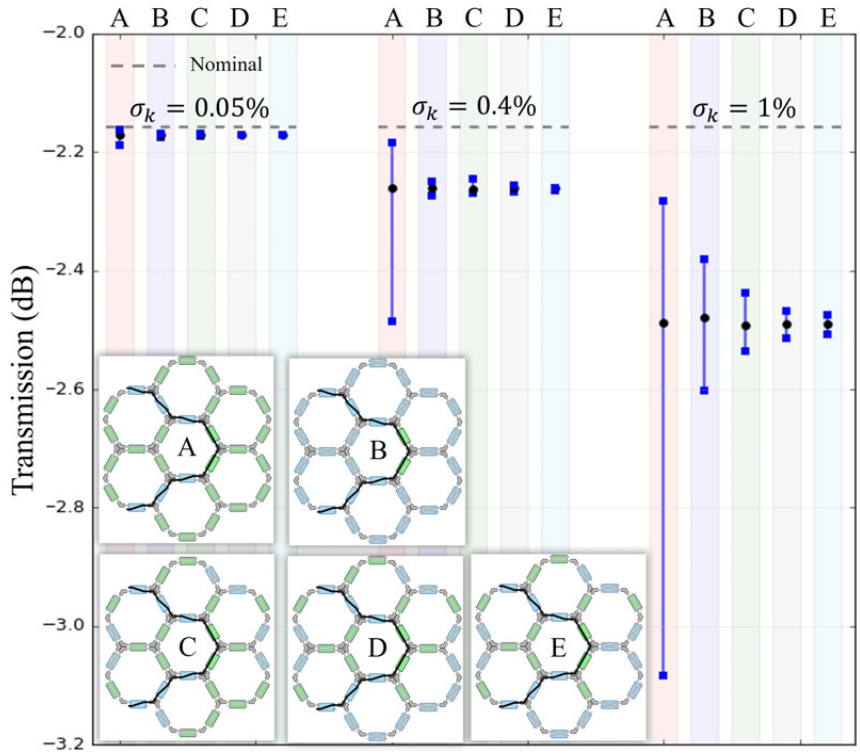


Figure 2.9: Transmission spread analysis of a single path (with $\Delta L = 8L_u$) for different biasing schemes for $\sigma_k = 0.05\%, 0.4\%, 1.0\%$.

2.2 Discretization Errors

As discussed in the previous section, there are various sources of error in the programmable meshes that all together will cause parasitic behaviour in the mesh and dramatically deteriorate the response of the circuit. In this section, we focus more specifically on discretization errors in controlling 2×2 tunable couplers which are the key elements (tunable units) in programmable PICs and can be arranged in a mesh of waveguides to create reconfigurable paths for the optical signals [22].

These tunable units operate either as an optical crossbar switch (in cross or bar) or as a tunable power divider. The common implementation consists of a Mach-Zehnder interferometer (MZI) with phase shifters in one or both arms [22, 50, 103]. By applying electrical signals the coupling between the input and output ports can be adjusted. The performance of a programmable circuit depends entirely on how accurate these coupling ratios can be controlled. Coupling errors can accumulate and propagate through the circuit, and result in optical losses and crosstalk.

In realistic integrated optical devices, various sources of error such as propagation losses, phase errors, and unbalanced beam splitters can severely impact performance of the tunable 2×2 couplers and consequently deteriorate the behavior of the circuit. Even though the tunability of the 2×2 couplers can compensate some fabrication errors, imperfect control of the phase shifters in the MZI may actually induce additional errors. As in any realistic system the phase shifters are controlled by some form of digital circuit such as a digital-to-analog converter (DAC), the digital discretization can be a source of errors. As the cost of DACs increases with increased resolution, it is important to understand how the resolution of digital drivers affects the coupling control so we can design and control the 2×2 couplers in a way that is tolerant to digital driving.

Here, we compare three different implementations (Fig. B.2) of thermally tunable (Sec. 2.2.1) and MEMS-based (Sec. 2.2.2) 2×2 MZI couplers, and analyze their coupling errors caused by digital voltage drivers with different resolutions (4-16 bits). These tunable couplers consist of two 3-dB couplers, and one or two phase shifters and can be reconfigured by digital voltage drivers to tune between bar state and cross state. **Type A** has equal arm lengths and is loaded with an actuator on one arm. **Type B** has an arm with $\pi/2$ phase delay (quadrature) and the MZI is loaded with actuators on both arms and operated in push-pull, i.e. only one of the actuators is used at any given time. And, **type C** is an MZI with a $\pi/2$ phase delay (quadrature) loaded with phase shifters on both arms. In type C, the phase shifters are used together creating discrete 2D-space for the coupling. Performing quantitative analysis, we show that proper biasing of couplers and simultaneous

driving of arms can improve discretization errors.

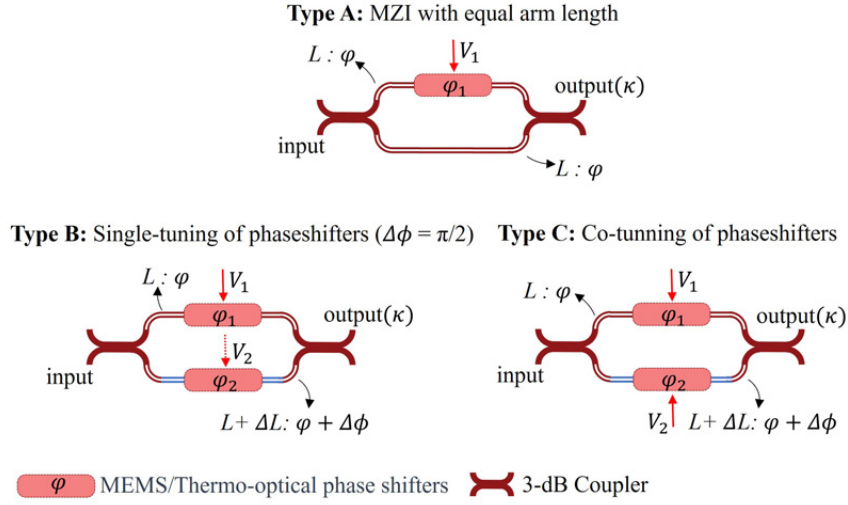


*Figure 2.10: Tunable 2 × 2 MZI couplers phase shifters. **Type A**: MZI with equal arm lengths is loaded with an actuator on one arm. **Type B**: MZI with a $\pi/2$ phase delay (quadrature) loaded with actuators on both arms. In type B, the coupler is operated in push-pull, where only one of the phase shifters operating at any given time. **Type C**: MZI with a $\pi/2$ phase delay (quadrature) loaded with heaters on both arms. In type C, the phase shifters are used together creating discrete 2D-space for the coupling.*

2.2.1 Thermally Tunable 2 × 2 MZI Couplers

Thermally tunable couplers use heaters as thermo-optic phase shifters which induce the desired phase shift by heating the waveguide with an electric current (Joule effect), making the phase shift proportional to the burnt electrical power in the heater [22, 50, 103]. For the simulations, we assume the heaters are voltage-controlled[1], so the phase shift can be written as:

$$\Phi_{PS} = \Phi_{full}.V^2/V_{full}^2 \tag{2.5}$$

where V_{full} is the voltage needed to induce the required full phase shift Φ_{full} in the corresponding waveguide arm. We assume that we operate with a driving voltage of 5V, and we designed the heater resistance to have a $V_{full} = 4V$.

The digital-to-analog converter will discretize the 0-5V voltage with n bits, resulting in 2^n voltage levels. This implies that we will get a corresponding discrete

[1]The reasoning can also be used for a current driver.

set of coupling κ values.

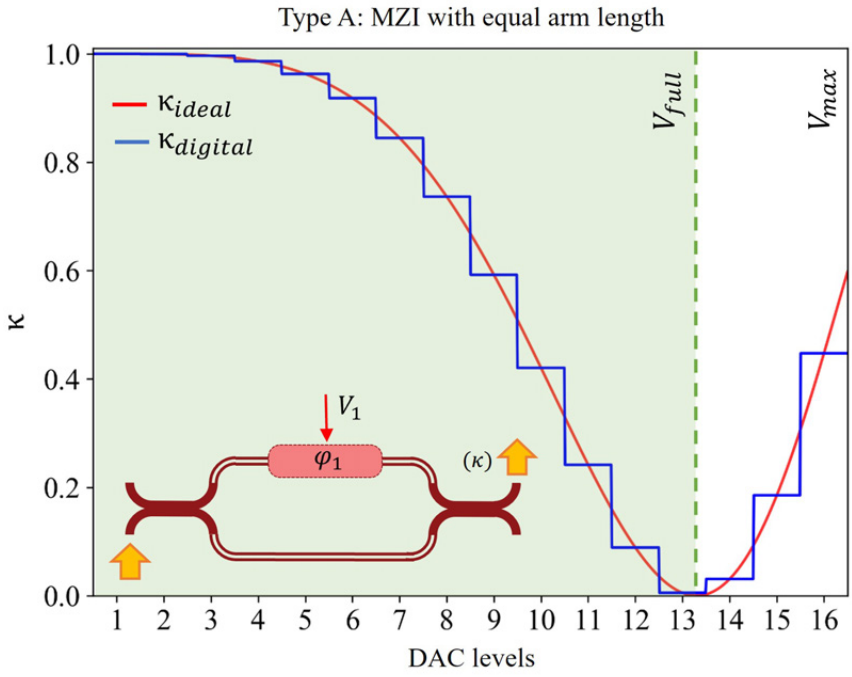


*Figure 2.11: Coupling response of the tunable 2 × 2 MZI coupler using thermo-optic phase shifters (**Type A**). MZI with equal arm lengths is loaded with a heater on one arm. A 4-bit DAC divides the input voltage of the heater into 16 levels, where $V_{max} = 5V$ and $V_{full} = 4V$.*

Fig. 2.11 shows the coupling κ response of the type A MZI coupler based on the levels of a 4-bits DAC (We use 4-bit resolution for illustration purposes, but the actual drivers will have a higher resolution). This MZI requires a $\Phi_{full} = \pi$ phase shift to couple from cross to bar state. The coupling response of the MZI shows that the discretization errors (indicated for 4-bit discretization for visual clarity) increase dramatically for larger voltages. This is because of the phase shifter's quadratic response, but also because of the sinusoidal response of the MZI. The steeper the slope of the response curve, the larger the discretisation error in the coupling value.

In contrast, Fig.2.12 shows coupling response of a MZI coupler with a $\pi/2$ phase delay between the arms, and a thermo-optic phase shifter in each arm. This biases the MZI at quadrature point: when both phase shifters are off, the MZI acts as a 50/50 beam splitter. Tuning one arm will decrease the coupling, while tuning the other will increase the coupling. In this configuration the full tuning range can be achieved with a $\Phi_{full} = \pi/2$ in either one or the other phase shifter.

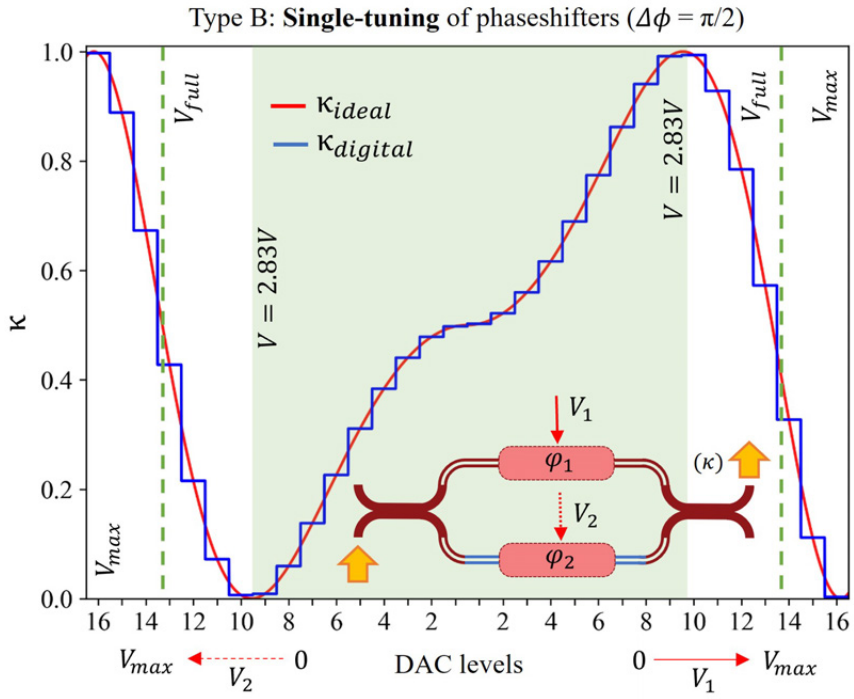


*Figure 2.12: Coupling response of the tunable 2 × 2 MZI coupler using thermo-optic phase shifters (**Type B**). Here, the MZI has a $\pi/2$ phase delay (quadrature) loaded with heaters on both arms. And, the coupler is operated in push-pull, where only one of the phase shifters operating at any given time. A 4-bit DAC divides the input voltage of each heater into 16 levels, where $V_{max} = 5V$ and $V_{full} = 4V$. As seen, the complete state change from bar to cross is supported by 20 DAC levels giving more accuracy compared to the type A.*

Also, the quadrature bias shifts the nonlinear heater response with respect to the sinusoidal MZI response, spreading the discretization error more uniformly across the coupling range. Another point to mention is the voltage range and the number of DAC levels available to switch from Bar state ($\kappa = 0.0$) to cross state ($\kappa = 1.0$). As illustrated in Fig. 2.11, type A undergoes a complete state change within a voltage range of 4V using 13 DAC levels. In contrast, type B achieves a full state change over a voltage range of 2.83V (for each phase shifter) with 20 DAC levels. This indicates that while type B offers higher control resolution, it also requires less power for each phase shifter to alter the coupler state.

Unlike type B, we operate both phase shifters together in type C. This co-tuning of the phase shifters, together with the nonlinear response of the heaters, creates a discrete 2D-space to control the coupling κ using the digital voltages V_1 and V_2. This 2D discretization increases the resolution of the tuning, with a voltage pair that

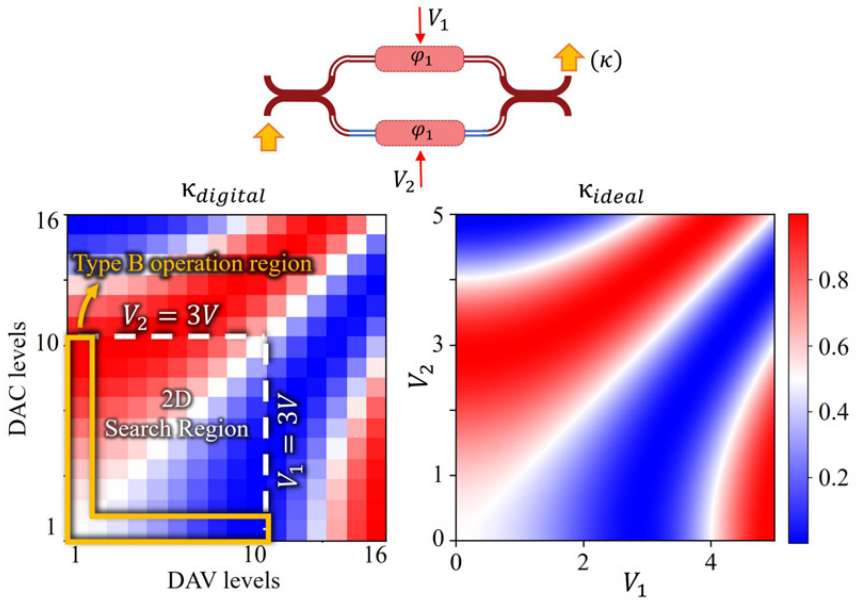


*Figure 2.13: Coupling response of the tunable 2 × 2 MZI couplers using thermo-optic phase shifters (**Type C**). In this arrangement, we have a MZI with a $\pi/2$ phase delay (quadrature) loaded with heaters on both arms. The phase shifters are used together creating discrete 2D-space for the coupling. Yellow region corresponds to the discrete response of type B, and the white box shows the 2D region to search the best voltage pairs for a desired coupling κ.*

brings the resulting $\kappa_{digital}$ closer to the desired value κ_{ideal}. The yellow region in Fig. 2.13 indicates the coupling levels of Type B as a small subset of the levels available in Type C. As we showed in type B, setting ($V_1 = 2.83V$, $V_2 = 0.0V$) and ($V_1 = 0.0V$, $V_2 = 2.83V$) gives us cross and bar state; hence, to find the optimum values of voltage in 2D voltage space of type C, we just need to search a small region as indicated by the white box in Fig. 2.13.

For further analysis, Fig. 2.14a plots the $\kappa_{digital}$ response of the three discussed couplers versus the ideal (desired) couplings for a 4-bits voltage driver. Comparing the curves clearly show that the MZI in quadrature increases the accuracy of the coupling selection for Type B. And co-tuning of the phase shifters considerably improves the performance of Type C. We quantified the maximum step size (σ_{max}) illustrated by the black arrow in Fig. 2.14a for each curve, and this for different resolutions of the DAC. Figure 2.14b shows the variation of (σ_{max}) versus different voltage resolutions (4-16 bits) for the three types. As expected, by increasing the number of bits, the maximum step size of the coupling for all 2×2 couplers decreases. And, interestingly, using the co-tuning scheme results in a noticeable

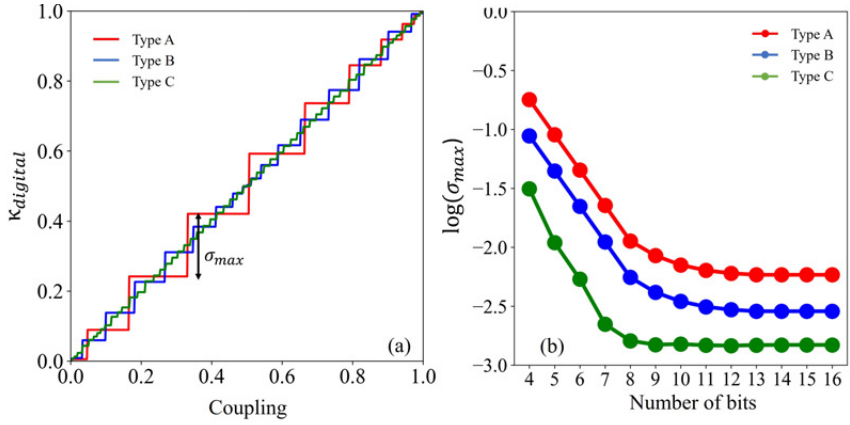


Figure 2.14: Left: Digital couplings κ of three different types of tunable couplers (type A, type B, and type C) versus desired ideal couplings for a 4-bits digital voltage driver. Right: comparison of maximum step size (σ_{max}) of the digital couplers for different resolution of 4-16 bits. Black arrow indicates σ_{max} of the type A.

reduction of σ_{max} in Type C, without needing to resort to more expensive electronics; for instance, even with low-resolution control (4 bits), σ_{max} is reduced from 18% in Type A to 3% in Type C (on a logarithmic scale, from -1.5 to -0.74).

Although co-tuning of two phase shifters significantly improves performance of a tunable coupler, it may have some downsides. For example, it increases the energy consumption or may complicate driving due to thermal cross-talk in the heaters, requiring larger separation and therefore larger footprints. On the other hand, coupling errors add up and adversely grow in large-scale circuits, as we have discussed at length in the previous section.

2.2.2 MEMS-based Tunable 2 $\times$ 2 MZI Couplers

We have also performed the above analysis for the MEMS-based MZI couplers, where we have used a MEMS phase shifter [10] designed by KTH as discussed in the first chapter. Figure 2.15a shows the measured phase shift of this components for the actuation voltages of 0V to 36V. As seen, for the 36V actuation we can have a phase shift close to 3π. Since the data points obtained by the measurements are not enough for our analysis, we have used a mathematical model (fitted curve in Fig. 2.15a) for the simulations.

The coupling response of the type A MZI coupler using this phase shifter is plotted in Fig. 2.15b, where the inset shows the full response when 3π phase shift

is applied. Similar to the heater-based coupler discussed in the previous section, we select the V_{max} to be slightly higher than the zero-coupling voltage (9.92V in this case); hence, the V_{max} will be 12V. As the plot illustrates, similar to the type A for heater-based coupler, the discretization errors (indicated for 4-bit discretization for visual clarity) increase dramatically for larger voltages.

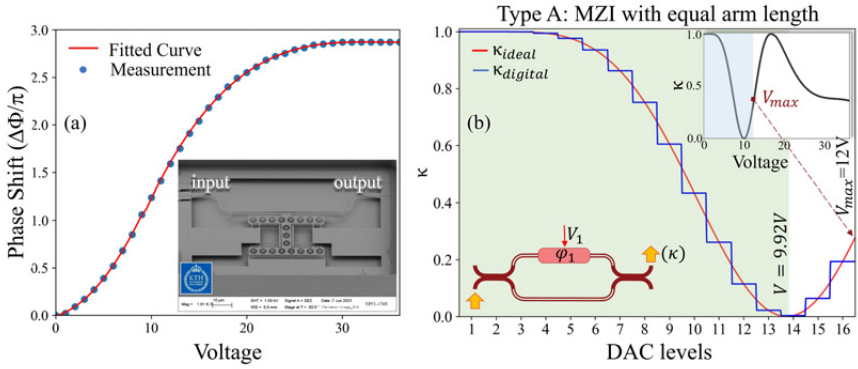


*Figure 2.15: a) Phase shift response of a MEMS phase shifter designed by KTH. b) The coupling response of the **type A** tunable 2 × 2 MZI couplers using the MEMS phase shifter. The MZI has equal arm lengths and is loaded with a phase shifter on one arm. The inset shows the full coupling response of the MZI coupler, while the main figure shows the response for $V_{max} = 12V$. A 4-bit DAC divides the input voltage of the MEMS into 16 levels.*

Next, the coupling responses of the MZI couplers for type B and C are shown in Fig. 2.16a and Fig. 2.16b, respectively. As expected, in type B, the quadrature bias shifts the nonlinear response of the MEMS phase shifter with respect to the sinusoidal MZI response; this spreads the discretization error more uniformly across the coupling range. Additionally, higher number of DAC levels are available and the voltage range for complete state change has been reduced. Figure 2.16b shows the 2D coupling response of the type C MZI coupler, where we have implemented the co-tuning of the phase shifters on both arms. Similar to the previous case we expect to improve the accuracy of our actuation.

Fig. 2.17a illustrates the $\kappa_{digital}$ response of the type A, type B, and type C MZI couplers versus the ideal (desired) couplings for a 4-bits voltage driver. Similar to the heater-based couplers, comparing the curves clearly shows that the MZI in quadrature increases the accuracy of the coupling selection for Type B. And co-tuning of the phase shifters considerably improves the performance of Type C. We quantified the maximum step size (σ_{max}) illustrated by the black arrow in Fig. 2.17a for each curve, and this for different resolutions of the DAC. Figure 2.17b shows the variation of (σ_{max}) versus different voltage resolutions (4-16 bits) for

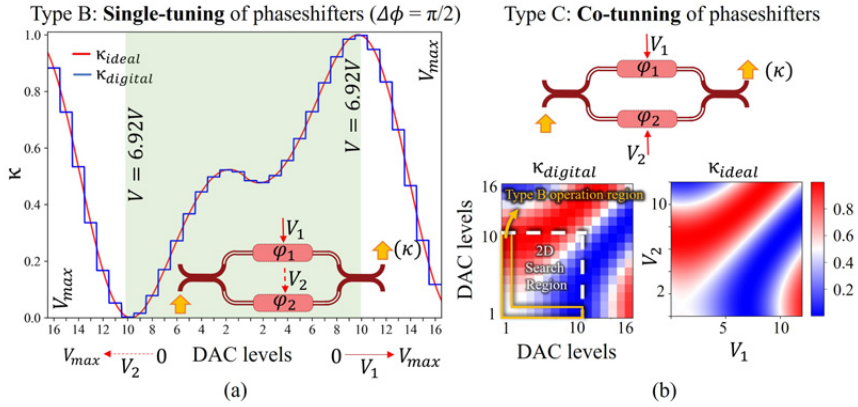


*Figure 2.16: Coupling response of the a) **Type B** and b) **Type C** tunable 2 × 2 MZI coupler using MEMS phase shifters. For type B, a 4-bit DAC divides the input voltage of each MEMS device into 16 levels, where $V_{max} = 12V$. As seen, the complete state change from bar to cross is supported by 20 DAC levels giving more accuracy compared to the type A. For type C, we have a MZI with a $\pi/2$ phase delay (quadrature) loaded with heaters on both arms. The phase shifters are used together creating discrete 2D-space for the coupling. Yellow region corresponds to the discrete response of type B, and the white box shows the 2D region to search the best voltage pairs for a desired coupling κ.*

the three types. As expected, by increasing number of bits, the maximum step size of the coupling for all 2×2 couplers decreases. And, co-tuning scheme noticeably reduces σ_{max} in Type C; for example, even with low-resolution control (4 bits), σ_{max} is reduced from 17% in Type A to 4% in Type C (on a logarithmic scale, from -0.76 to -1.39).

One important point to mention is that the discussed scheme relies very much on the fact that the phase shifter has a nonlinear response curve. Without this nonlinearity, the scheme wouldn't work, as one phase shifter would counteract the other. While we usually aim for a linear response to simplify design, this demonstrates that a nonlinear response can offer significant benefits.

2.3 Summary

In the first part of this chapter, two biasing schemes for the unused couplers in a hexagonal 7-cell mesh were compared for their impact on different configurations of single paths, multipaths, ring resonators, MZIs, and splitters. In these schemes, the unused couplers were set either in the bar state (NB bias) or cross state (NC bias). Monte Carlo simulations showed that NC bias considerably suppresses the

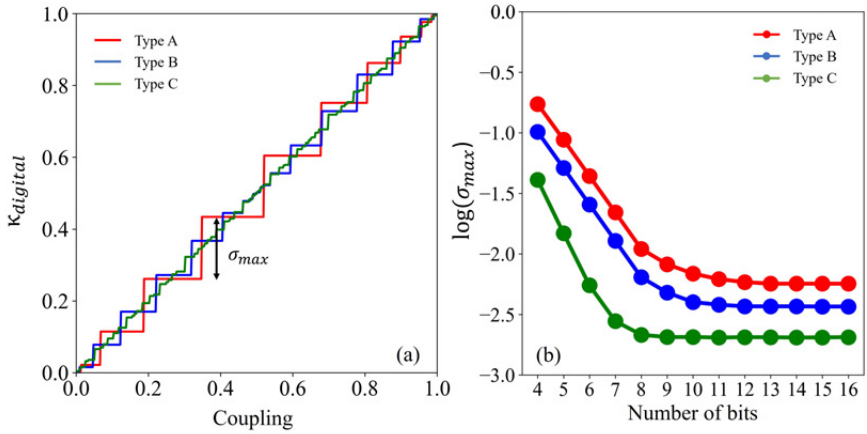


Figure 2.17: Left: Digital couplings κ of three different types of MEMS-based 2×2 tunable couplers (type A, type B, and type C) versus desired ideal couplings for a 4-bits digital voltage driver. Right: comparison of maximum step size (σ_{max}) of the digital couplers for different resolution of 4-16 bits. Black arrow indicates σ_{max} of the type A.

effects of parasitics on the transmission response of the mesh for straight paths and becomes less effective for complex paths with loops. The transmission spread of the paths exponentially grows with increase of the coupling errors; further, it can vary from 0.03 dB ($\sigma_\kappa = 0.05\%$) to 1.8 dB ($\sigma_\kappa = 1.0\%$). Using less than four couplers out of six per cell more or less guarantees the possibility to compensate the parasitics using NC bias. Paths with loops in the center of the mesh are more vulnerable to parasitics compared to those with loops near edges of the mesh. Also, using custom optimized biasing, we have shown that it is also possible to eliminate parasitic effects with fewer unused couplers.

On the other hand, for ring resonators and MZIs in which the phase shifter's error is dominant and affects the depth of nulls, both biasing schemes have similar effects. However, selecting the ring near the edges of the mesh can improve correlation of the perturbed response with the ideal one. Moreover, the example of the 1×16 beam splitter shows that, although it is possible to use most of the mesh couplers to achieve a certain configuration, it limits our freedom to compensate parasitics by the remaining unused couplers.

It is important to realize that our compensation mechanism relies on shunting all the unused light to the edge of the circuit. When the mesh grows larger, there is less "edge" compared with "bulk." Therefore, a large mesh could benefit from "beam dumps" in different places inside the mesh (maybe even each cell), which could be activated by switches. The use of switches could introduce another benefit: in these

simulations, we used tunable couplers, which are controlled in a continuous way; in fact, in most locations they are used as switches in a cross or bar state. Using a mix of digital switches and tunable couplers in a mesh might help suppress parasitics, if the switches can have a better guaranteed extinction ratio.

Overall, we can say that more precise control results in lower parasitics. This requires good electronics, calibration, and control of other parasitics (electronic and thermal crosstalk). Incorporating control loops in a recirculating mesh (just like in feed-forward meshes) is also possible but not without additional tricks, such as the use of power monitors inside the mesh (e.g., CLIPPs [25]). Depending on the combinations of light paths in the mesh, we would require some form of labeling of the signals (e.g., with a frequency pilot tone), so we can know which light is inadvertently coupled to the wrong path [104]. As a main conclusion, we can state that recirculating meshes will suffer from parasitics, more than forward-only meshes, but that over dimensioning the mesh and good biasing of the unused couplers can go a long way in suppressing unwanted resonances that these parasitic couplings will generate.

In the second part of this chapter, We discussed the effect of discretized voltage driving in three different implementations of tunable $2\times$ 2 MZI couplers with thermo-optic and MEMS-based phase shifters. The discrete voltage response of the digital drivers causes a staircase error, resulting in a nonuniform discrete coupling response of the MZIs. Simulation results reveal that using an MZI in quadrature with two phase shifters, and co-tuning of both phase shifters, can significantly reduce the discretization error.

3

Mesh Analysis and Circuit Design

In this chapter, we first analyze various mesh shapes and sizes. Next, we discuss the final Field-Programmable Photonic Integrated Circuits (FP-PICs) designs fabricated in MORPHIC RUN2 and RUN3, along with the switch circuits used for our characterizations and debugging process.

This work has been published in [105].

3.1 Mesh Analysis

Key decision parameters in designing the mesh of a programmable photonic circuit include *topology*, *shape*, *loss*, and *scaling*. Several factors significantly influence this process: the mesh's loss, the size and function of the circuits to be programmed within the mesh, and the limitations related to fabrication and packaging. In this section, we present the various mesh analyses we conducted before designing the FP-PIC circuits for MORPHIC.

3.1.1 Topology and Shapes

As discussed in the introduction chapter, meshes with hexagonal topology have highest flexibility in terms of routing, use of clockwise/counterclockwise, and choice of incremental delay lines [99]. Hence, we have selected the hexagonal topology for our mesh designs. However, for further evaluation, we also use three demonstrator cases to see if they can be implemented using hexagonal topology. The selected demonstrators are: an optical beamformer, a switch matrix, and a microwave photonic filtering circuit. The criteria for the mesh can be inferred from the smallest mesh that can host each demonstration circuit, i.e., a 1×8 beamformer, a $4\times4/6\times6$ switch matrix or a complete double ring loaded MZI (but not all together).

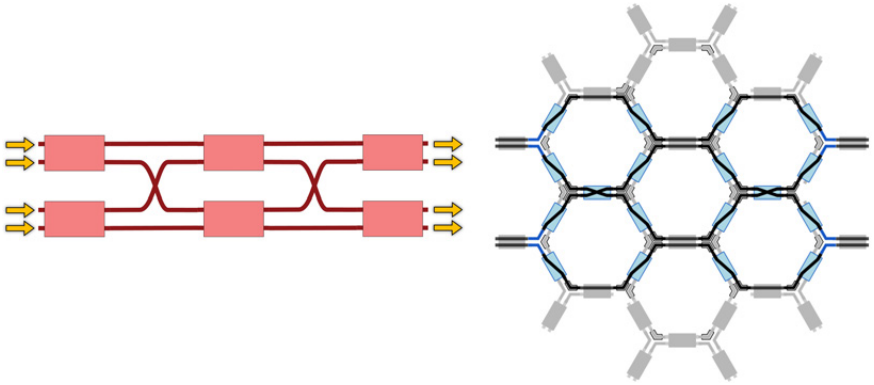


Figure 3.1: Translations of an ASPIC circuit for a Benes 4×4 switch, and its implementation in a hexagonal mesh.

Figure 3.1 shows a possible programming strategy implementing a Benes 4×4 switch. As switches can be implemented in different architectures (Benes, Crossbar, PI-loss) [106], the programming can be influenced by the chosen switch network. Benes offers the topology with the lowest component count when implemented as an ASPIC.

Figure 3.2 shows two possible implementations for a power distribution network for an optical phased array. This can be done through a bus waveguide from which power is tapped at regular intervals. Or it can be in the form of a tree network. Note that it is not trivial to implement these circuits in a mesh in such a way that all the path length differences are properly controlled, as a beamformer is phase sensitive and ideally all path lengths should be the same. Also, in a real mesh design, only certain ports will be coupled to the optical phase array (OPA) antennas. This means that we need to define a programming strategy for the mesh together with the chip design, not afterwards.

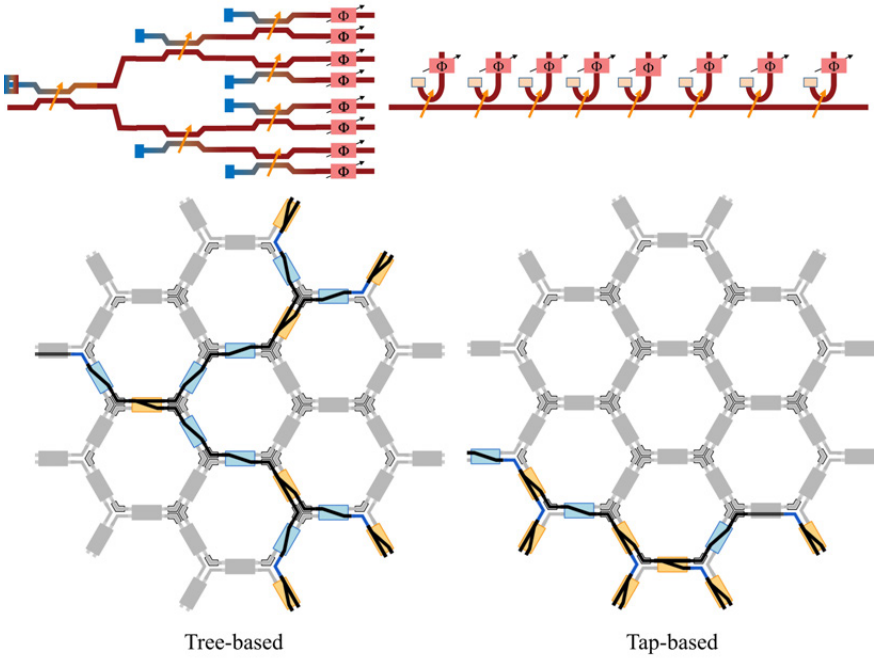


Figure 3.2: Implementations of a 1×8 distribution circuit for an optical beamforming network, and how they could be implemented in a programmable mesh.

Phase sensitivity is even more important in microwave photonics filters. Here, the mesh can be used to implement ring resonators and interferometers, and path length control is even more important. Figure 3.3 shows an MZI filter with 4 rings that implements a double ring arm loaded MZI filter.

The systems presented in Fig. 3.1, Fig. 3.2, and Fig. 3.3 are possible implementations on a mesh with hexagonal topology. On a small scale, these are easy to implement manually, but as they scale up, it becomes harder to program them. Also, it becomes difficult to estimate the minimum mesh size and shape that is needed to implement each demonstrator. In order to determine the maximum viable mesh size, different mesh shapes and topologies should be considered.

Here we use hexagonal unit cells and arrange them into different mesh outline shapes which can be categorized as convex and non-convex. A mesh is considered *convex* if, for any two points within the mesh, the line segment connecting them lies entirely within the mesh. This implies that there are no indentations or holes in the mesh, and the shape is ”bulging outwards.” Mathematically, a mesh is convex if every interior angle is less than or equal to 180 degrees. A mesh is *non-convex* if there exists at least one pair of points within the mesh such that the line segment

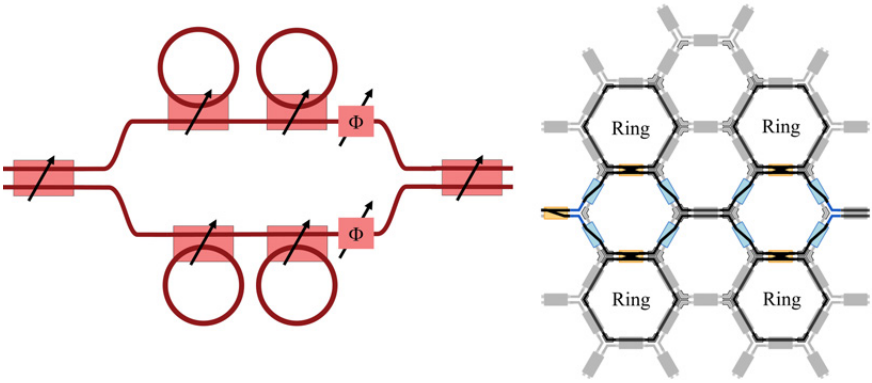


Figure 3.3: Translation of a 4-ring filter ASPIC for microwave filtering to a programmable mesh.

connecting them lies partially outside the mesh. This means that the shape can have indentations or be concave. Mathematically, a mesh is non-convex if at least one interior angle is greater than 180 degrees. Figure 3.4 shows the abstract schematic comparison of convex and non-convex meshes.

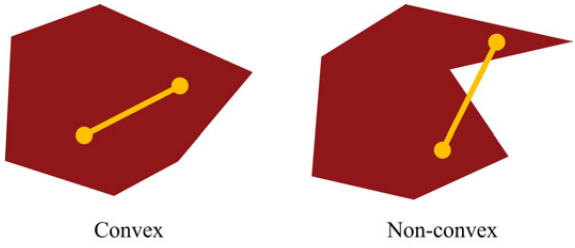


Figure 3.4: Schematic demonstration of convex and non-convex meshes.

For convex meshes, we have studied radial and rectangular meshes and, for non-convex meshes we have studied star and "fox" meshes. Short description of these shapes are presented in the following:

- **Radial mesh**: Having a hexagonal cell as the center of the mesh, we start adding cells in all directions and grow our mesh layer by layer. Here, we define r as the expansion parameter of the mesh. Fig. 3.5 shows three radial meshes for $r = 0, 1, 2$.
- **Square mesh** or **Rectangular mesh**: In a rectangular mesh we expand the periodic lattice along the X and Y direction, and therefore we have two expansion parameters (a, b). A mesh with hexagonal cells grows anisotropically

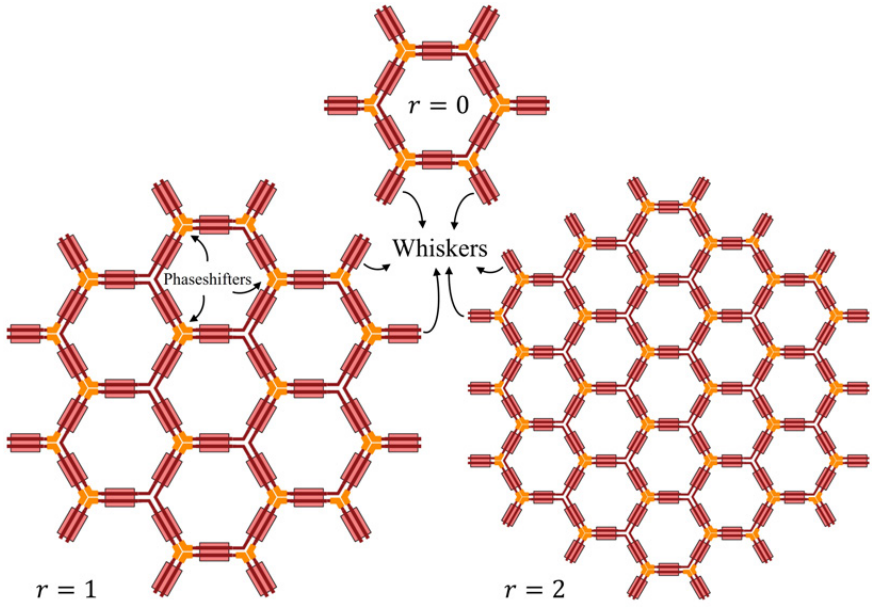


Figure 3.5: Radial meshes with hexagonal cells, where expansion parameters are r = 0, 1, 2. Note that each cell has three phase shifters.

for these directions. As a special case of rectangular mesh, the square mesh is constructed by increasing number the of cells in both x and y direction equally. Figure 3.6 shows two square meshes with expansion parameters of $(a, a) = (2, 2), (6, 4)$.

- **Star mesh**: Star mesh (Figure 3.7) is created based on the radial mesh and grows in all 6 directions which creates a star shape by adding extra triangular 'points'. This has the advantage of creating a larger perimeter with more interfaces.

- **Fox mesh**: Here we grow the mesh asymmetrically. The name 'Fox' alludes to the pointy 'ears' at the top, as shown in Figure 3.8. This gives at the same time areas of the mesh close to an edge, while still having larger contiguous areas that can be programmed for e.g. filter functions.

It should be noted that all these meshes have internally the same uniform hexagonal unit cell. This is not a strict requirement, and more diverse mesh topologies are possible. For example, we can combine hexagonal cells with square, triangular, or irrequlary shaped unit cells. Or, we can implement meshes with "holes", where additional functional blocks can be connected.

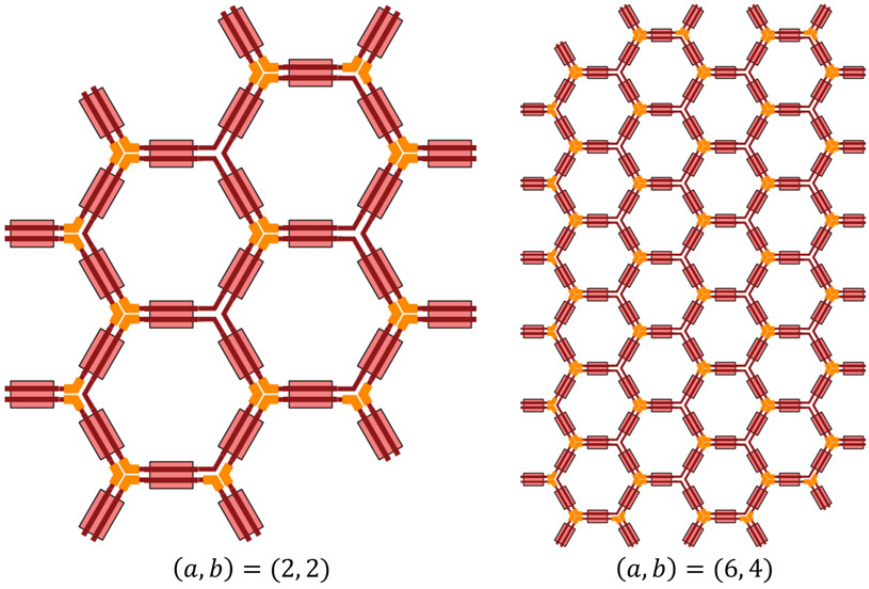


Figure 3.6: Rectangular meshes with hexagonal cells, where expansion parameters are (a, b) = (2, 2), (6, 4). Note that each cell has three phase shifters.

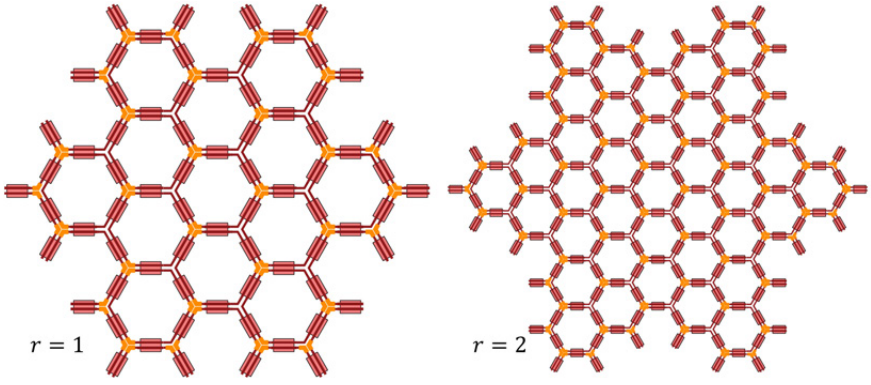


Figure 3.7: Star meshes with hexagonal cells, where expansion parameters are r = 1, 2. Note that each cell has three phase shifters.

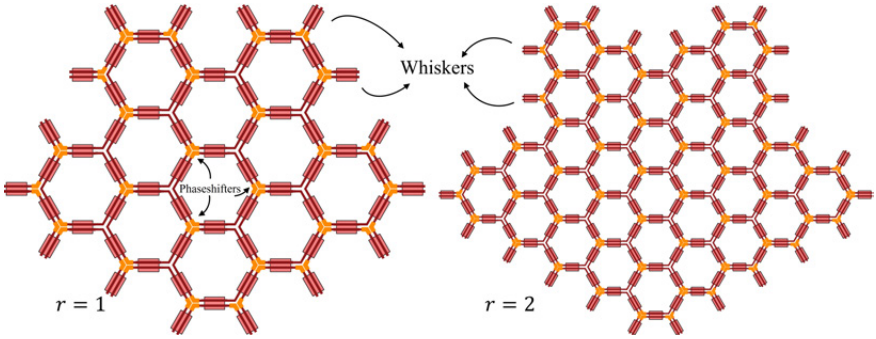


Figure 3.8: Fox meshes with hexagonal cells, where expansion parameters are r = 1, 2. Note that each cell has three phase shifters.

In uniform hexagonal meshes, the expansion parameter (or scaling factor) relates to the ratio between the periphery and the mesh area. To compare selected mesh blocks when we scale them, we have performed a quantitative analysis on several mesh parameters:

- Number of couplers and phase shifters.
- Number of 'whiskers' (i.e. the number of freestanding couplers on the edge of each mesh).
- Number of I/Os (optical ports).
- Number of DC control connections (related to number of phase shifters and couplers).
- The ratio of the whiskers to all couplers.

In Fig. 3.9, these values have been summarized and plotted versus the number of cells in each mesh. It should be mentioned that these meshes have been constructed based on the cells where we have only three phase shifters per cell, as opposed to 6 cells in a fully regular mesh. It is obvious that, by expanding the mesh and thus increasing number of cells, the number of phase shifters, couplers, and consequently the number of DC connections increases almost linearly; however, the number of ports (each whisker has two ports) has a sublinear relation with number of cells. This can be intuitively understood: the number of cells is proportional to the edge-to-area ratio of the mesh, while the number of whiskers is proportional only to the length of the edge. Depending on the shape of the mesh, the edge-to-area ratio will vary. Radial meshes, because of the convex nature of their mesh boundary, have the lowest number of I/O ports for a given count of cells.

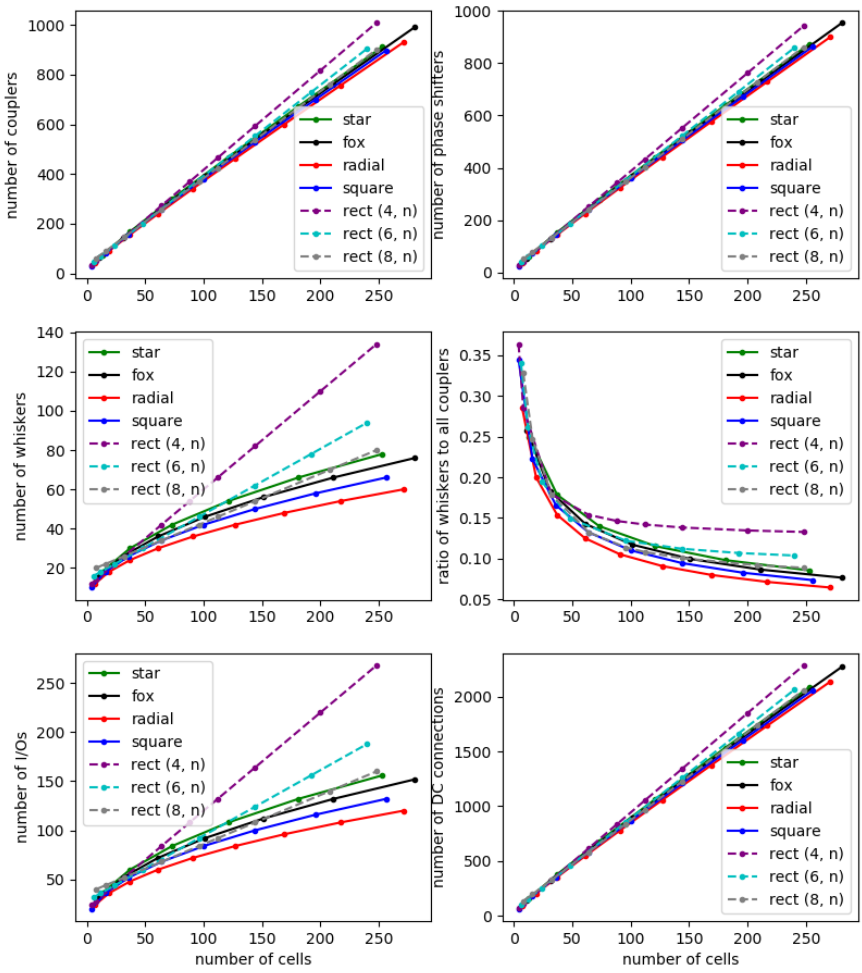


Figure 3.9: A comparison of properties of different simulated mesh shapes with hexagonal unit cell.

3.1.2 Average Path Loss of the Mesh

One of the important parameters in mesh scaling is loss. Here, we consider two scenarios: a) The average path loss of the mesh, and b) The loss of the longest shortest path in the mesh.

We also are interested to see how the loss of the components affect mentioned losses in the mesh; for this, we vary loss values of the waveguides to see how large our meshes can grow. To compare average path loss of meshes, we select

the convex meshes which are the radial, square, and rectangular meshes. The corresponding mesh expansion parameters are chosen as shown in Table 3.1. To calculate average path loss, loss of the shortest path between all possible port-pairs have been calculated and divided by the number of all port pairs. The calculation of the path loss is performed by calculating the shortest possible path between all combinations of ports in the mesh.

In a convex mesh, the shortest path between two nodes can be determined in sublinear time relative to the number of nodes in the mesh. This means that as the number of nodes increases, the time it takes to find the shortest path grows at a rate slower than the total number of nodes. This efficiency is due to the fact that, in a convex mesh, the structure allows for direct or nearly direct routing options that reduce the computational complexity of the pathfinding algorithm. This is particularly beneficial in photonic circuits, where quick and efficient routing is essential for maintaining high-speed data transmission and processing.

In contrast, a non-convex mesh lacks this structural simplicity. In such meshes, the shortest path between two nodes may involve detours or complex routing that navigates around obstacles or non-convex regions within the mesh. As a result, the time required to compute the shortest path can scale polynomially with the number of nodes. This means that as the number of nodes increases, the time required to find the shortest path can grow very rapidly, making it inefficient for large networks. The increased complexity in non-convex meshes arises from the need to evaluate a significantly larger set of potential paths, some of which may be far from optimal, before determining the shortest one.

Table 3.1: Expansion parameters of the radial, square, and rectangular meshes used for average path loss calculations.

Mesh Name	**Expansion Prameter**
Radial	$r = [1, 2, 3, 4, 5, 6, 7]$
Square	$(a, a) = [(2, 2), (4, 4), (6, 6), (8, 8), (10, 10), (12, 12)]$
Rect (2, n)	$(a, b) = [(2, 1), (2, 2), (2, 8), (2, 16), (2, 22), (2, 28), (2, 36), (2, 50)]$
Rect (4, n)	$(a, b) = [(4, 1), (4, 2), (4, 8), (4, 16), (4, 22), (4, 28), (4, 36)]$
Rect (6, n)	$(a, b) = [(6, 1), (6, 2), (6, 8), (6, 16), (6, 24)]$

Figure 3.10a-b show simulation results of the *number of all possible port pairs* and the *average path loss* of the selected meshes. As seen, by moving from a radial mesh to the rectangular meshes, the number of port-pairs rapidly increases. This is related to their respective edge-to-area ratios. This property also affects the average path loss as shown in Figure 3.10b. The loss calculations are based on the component losses measured on the photonic MEMS devices from MORPHIC RUN2, which amounts to 0.75 dB loss per segment; each segment includes loss of

a coupler/phase shifter and one transition from air clad to oxide clad (see section 3.2.3.1 for transitions).

The radial mesh has a lower average path loss, which is logical: As the mesh shape approaches a circle, the average path length gets smaller. The average path loss is a good measure for the losses that will be incurred during multi -routing (i.e. the simultaneous routing of many port pairs). While the radial mesh scales better in terms of losses, it may impose restrictions for some applications, which will be discussed later. Meshes with a higher edge-to-area ratio have a worse scaling behaviour in terms of average path loss. For further analysis we also performed average path loss analysis for the rectangular meshes with expansion parameters varying from $(2, 2)$ to $(18, 18)$, as shown in Figure 3.10c.

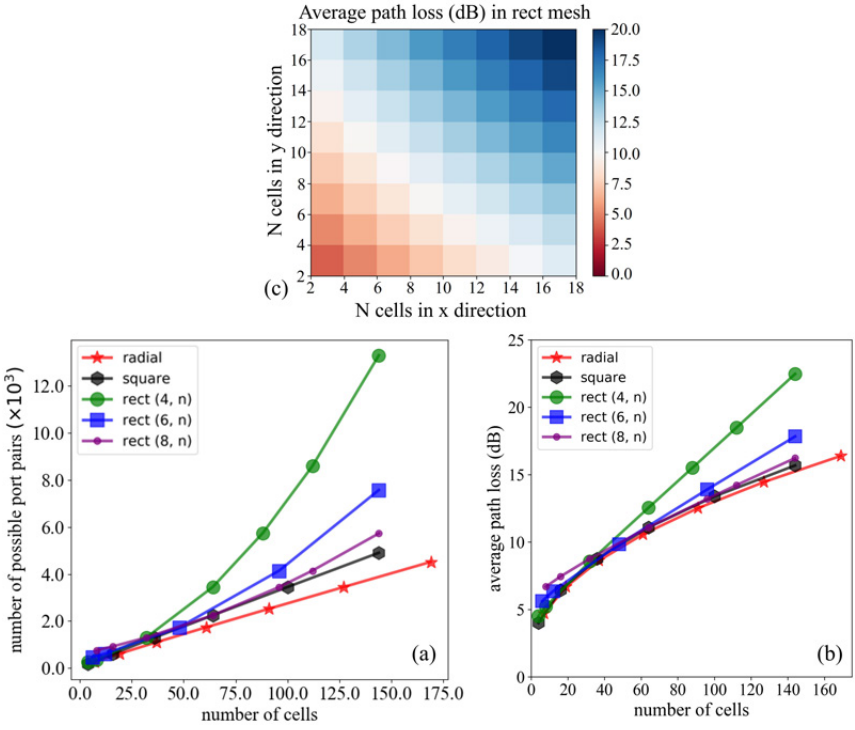


Figure 3.10: a) Number of the all possible port-pairs in radial, square, and rectangular meshes. b) Average path loss for each mesh as a function of the number of cells. This is calculated averaging the path losses from one input to each output. c) Loss a rectangular topology as a function of the number of cells in x and y direction.

3.1.3 Loss of the longest shortest-path

To study routing behavior, a direct connection from one port to the furthest I/O of the mesh is considered. We call this the "longest shortest path" which in graph theory is referred to as *radius* of a graph and, for a single port, we call this the *eccentricity*. Figure 3.11a shows an example of a shortest path between two selected ports of a radial mesh with $r = 1$. For the simulations we select the radial, square, and rectangular meshes and sweep their expansion parameters, then we find longest path among all possible shortest paths in the mesh and calculate its loss. Similar to the previous section, the segment loss is estimated as 0.75 dB. Loss values plotted in Fig. 3.11b shows that for the given number of cells radial mesh has best loss performance which is in accordance with results of the average path loss shown in Fig. 3.10b.

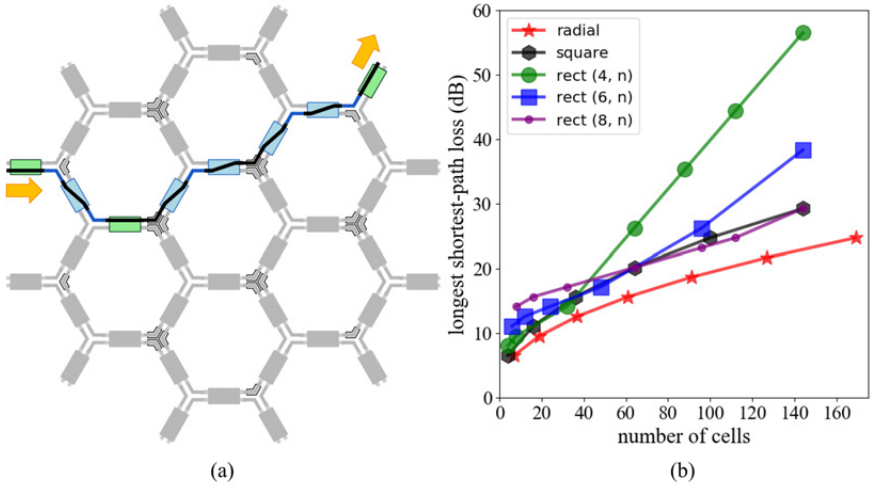


Figure 3.11: a) Path through the radial mesh with $r = 1$ considered for the evaluation of insertion loss. b) Longest shortest-path loss for different mesh shapes.

For further elaboration, We consider four scenarios:

a) Reference data: Segment loss of $0.75\,\mathrm{dB}$, where each segment includes air clad waveguides ($loss$ = $5.74\,\mathrm{dB/cm}$ and $length = 400\,\mu\mathrm{m}$) and oxide clad waveguides ($loss$ = $2.0\,\mathrm{dB/cm}$ and $length = 920\,\mu\mathrm{m}$).

b) $1.0\,\mathrm{dB/cm}$ decrease of waveguides losses.

c) Having no transition loss (equivalent to putting all circuit in a one large MEMS cavity).

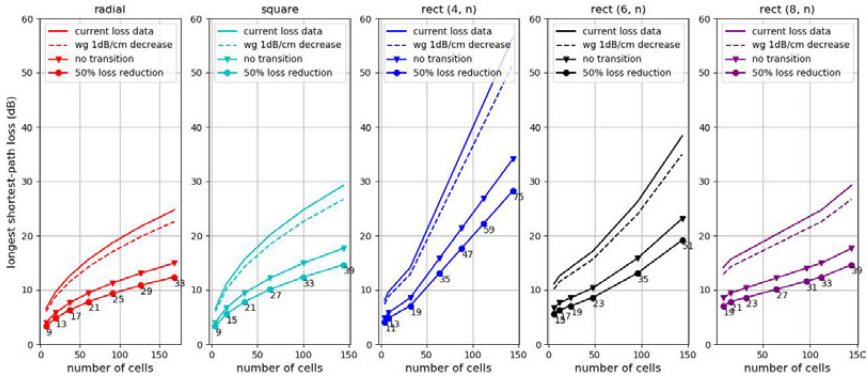


Figure 3.12: Longest shortest-path loss variation for radial, square, and rectangular meshes. Four cases have been considered: solid curves: using reference data (segment loss = 0.75 dB), dashed-curves: 1 dB/cm decrease of waveguides losses, solid-curves with triangle markers: eliminating all transitions, and solid-curves with circular markers: reducing all losses to the 50% of the current values.

d) Reducing all losses by $50\,\%$ of their current value.

The losses of the longest shortest path in the different mesh shapes are plotted for those four scenarios in Fig. 3.12. We see that, as expected, the losses go down, and for the expected improvements we get losses well below $20.0\,\mathrm{dB}$ for radial and square meshes.

3.1.4 Scaling Beam Splitter networks

Power distribution networks are an essential part of an optical beamforming system. As already illustrated in Fig. 3.2, such a network can be implemented as a bus with taps or a tree, or even a combination of both. We analysed how large a mesh with a certain shape needs to be to implement a tap-based or tree-based beam splitter network. Figure 3.13 shows a tree and tap-based networks in both a rectangular and a radial mesh. Also, we have plotted the number of cells needed to create a beam splitter with a given number of outputs, for different mesh shapes. We see that the tap-based network is much more scalable, as it can run along the periphery. For this reason, a thin rectangular network, with a large edge-to-area ratio, performs best in this metric. For a tree-based network, the number of cells needed rapidly explodes.

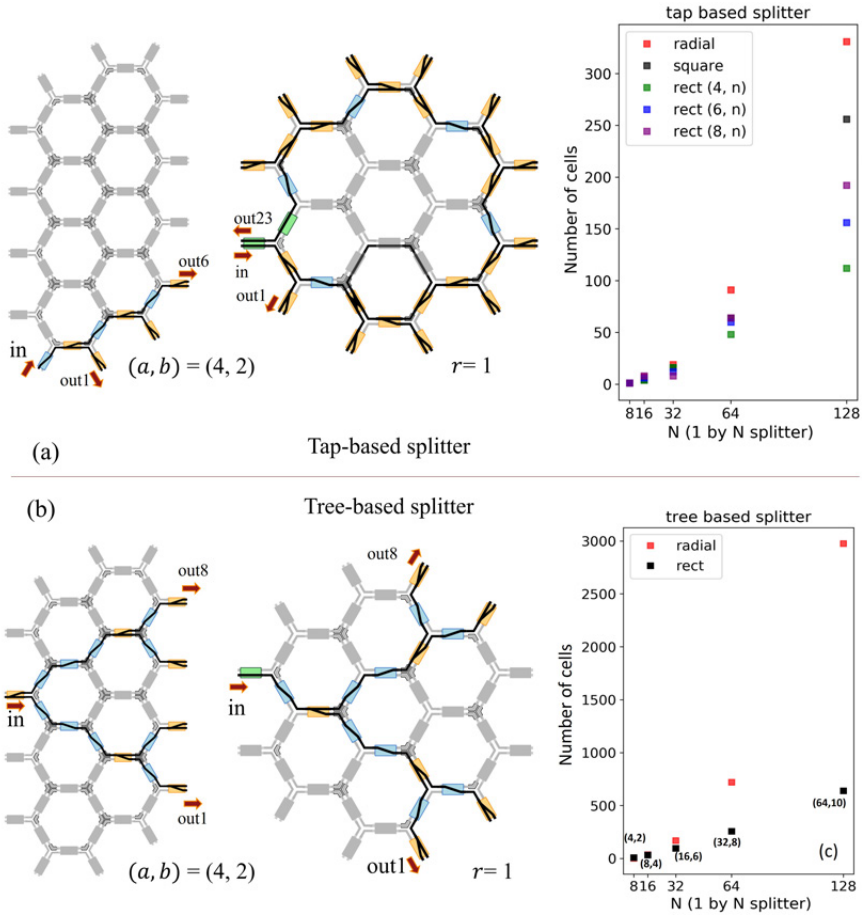


Figure 3.13: Schematics of a 1×8 tree-based beam splitter in rectangular and radial meshes. Similarly, a 1×23 tap-based beam splitter in a radial mesh ($r = 2$). The number of cells of different meshes required to (b) configure a tap-based $1 \times N$ beam splitter in a radial, square, and three different rectangular meshes. (c) number cells needed to build a tree-based $1 \times N$ beam splitter in a radial and rectangular meshes with $(a, b) = (4, 2), (8, 4), (16, 6), (32, 8), (64, 10)$.

3.1.5 Scaling of Switches

To analyze the performance of the mesh for N×N switching, we looked at the minimal mesh size needed to switch N ports on one side to N ports on the other side of the mesh in all possible permutations of ports. This being a first approximation. we did not investigate irregular arrangements of ports. The programmable mesh with a rectangular shape scales up in both directions (a, b) to enable N×N switching. In Fig. 3.14, we provide an example graph representation for a rectangular mesh with hexagonal cells and expansion parameters $(a, b) = (8, 7)$. The degrees of freedom for the choice of input-output ports of a switch circuit is large, such that any input can be connected to any output on a one-to-one basis. It quickly becomes computationally expensive to route all these possible pair combinations.

Thus, we assume that if a mesh can allocate N input-output pairs in both full-parallel and full-diagonal assignment, it can probably allocate the full N×N switch circuit. For example, Fig. 3.14a-b shows simultaneous multi-routing for 8 input-output pairs in parallel and in diagonal assignment, with a sparse assignment of inputs and outputs (one per whisker). We observe that every path will cross 7 other paths in Fig. 3.14b, so as long as we reserve sufficient crossing couplers, we can reduce the cell number a in the vertical direction, as shown in Fig. 3.14c. For a given number of inputs/outputs, we use a heuristic-based congestion negotiation algorithm (originally developed by my colleague Xiangfeng Chen [82, 107]) to sweep the values of both a and b. These simulations verify that the minimal size mesh for an N×N switch from left to right has boundary conditions as the mesh scales up: When b is an odd number as in Fig. 3.14c, the East side is symmetric with the West side, whereas when b is an even number as in Fig. 3.14d, the East side has more output ports than the West side. We choose the symmetrical configuration as the minimal requirement: solutions are found when $b \geq N - 1$ (b odd) and $a \geq N/2$. When b is even, the conditions are $b \geq N - 2$ and $a \geq N/2$. If the mesh is smaller than these conditions, generic left-to-right switching is no longer guaranteed.

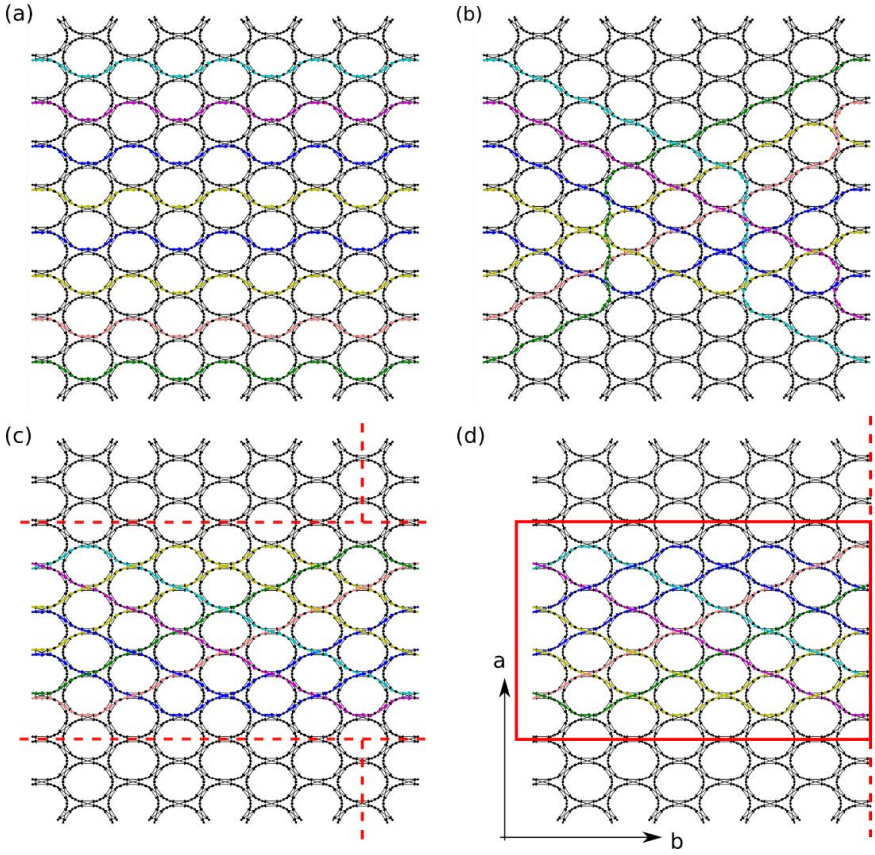


Figure 3.14: Graph-based routing in a rectangular-shaped mesh with 7×8 hexagonal cells. (a) Multi-routing finding the shortest paths routing for 8 input-output pairs in parallel and (b) in diagonal assignment. (b) shows a sparse use of 16 input-output ports (one per whisker) while (c) shows a dense use (two per whisker). The boundaries in dashed red lines shows the reduction in required mesh size, which is further improved in (d), representing the minimum mesh size to implement 8×8 switching.

3.2 Circuit Designs

The key objective of this work, as part of MORPHIC project, is to design and measure a generic field-programmable photonic integrated circuit (FP-PIC) which can perform many different functions [18]. It is expected that FP-PIC circuits can at least perform the same functions as the ASPIC demonstrators, although with some reduced performance (e.g. higher insertion loss due to the larger number of couplers).

As our preliminary designs, two small-scale FP-PIC demonstrators were implemented and fabricated on MORPHIC RUN2. The initial plan was to design 7-cell hexagonal meshes based on heaters and MEMS which could give us good comparison cases. However, during the design process we noticed that we could position more couplers and phase shifters in the PIC unit cells of MEMS mesh compared to the heaters circuit. As a results, the MEMS-based circuit has a 24-cell hexagonal mesh (Fig. 3.25), while heater-based circuit has 7 cells (Fig. 3.16) which also incorporates cavities to embed semiconductor optical amplifiers (SOA) using transfer printing. Because the amplifiers are large ($1.4\,\mathrm{mm}$ long), only a single hexagonal mesh cell can be fitted in a unit cell of the packaging grid. In RUN3, we aimed for a more ambitious design and implemented a FP-PIC with 126 hexagonal mesh cells. To avoid confusion, it is important to note that in an N-cell circuit, "N" refers to the number of hexagonal cells in the mesh, not the number of PIC cells on the chip.

First, the concept of the PIC unit cell created by the packaging bondpad grid is explained and compared with mesh cells. Next, FP-PIC designs are discussed in detail. And, finally, switch circuits are shortly reviewed which are useful for the measurement discussion in Chapter 6. Since the major focus of this thesis is on MEMS-based circuits, for simplicity, we will not use the terms of "MEMS-based" and "heater-based" when referring the FP-PIC circuits and meshes unless they are heater-based.

3.2.1 PIC Unit Cell

Before diving into circuit design details, it is important to clarify the terminologies and definitions used in our design process.

- **Mesh Cell**: It refers to the hexagonal cell (Fig. 3.15a) in the mesh representation of the programmable photonic circuit. As it will be discussed in Chapter 5, users can interact with the mesh to perform circuit simulations and configure their circuit. In fact, working with the mesh schematic is much

easier compared to inspecting the actual circuit layout.

- **Node:** It is a set of couplers and phase shifters based on hexagonal coordinates (Fig. 3.15a). In fact, nodes are abstract blocks of the mesh which can be used for circuit simulation, routing, and graph-based computations (see Chapter 5). We have two types of nodes: full nodes and custom nodes. *Full nodes* have the maximum number of couplers and phase shifters and consist of three couplers and three phase shifters. In contrast, *custom nodes* have only some of the couplers and phase shifters.

- **PIC cell** is a unit cell (Fig.3.15b) defined within our bondpad grid on the photonic chip. The grid system serves as an interface to interposers and is based on packaging strategies. For more details, refer to Chapter 4. Each PIC cell can accommodate up to two full-nodes (or 12 actuators).

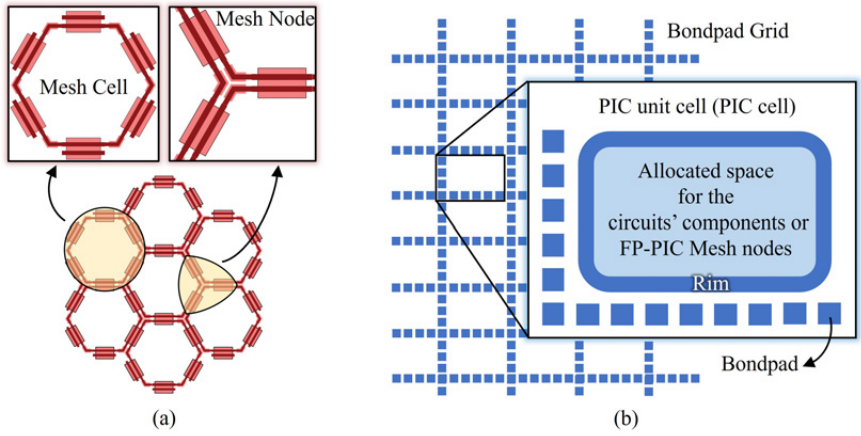


Figure 3.15: Schematic illustration of a) 7-cell hexagonal mesh highlighting the mesh cell and the mesh node. And, b) PIC unit cell defined on the bondpad grid implemented on MORPHIC chips.

3.2.2 7-cell FP-PIC (heater-based)

As shown in Fig. 3.16, the heater-based circuit is based on the 7-cell hexagonal mesh connected to the 'whiskers' (additional couplers at the corners which are meant to improve flexibility of the original mesh). This mesh is constructed by connecting 42 2×2-couplers (CP) and 48 phase shifters (PS) using waveguides. We have adjusted the topology of the original mesh by introducing some asymmetry to incorporate semiconductor optical amplifiers (SOAs). For scripting the layout

design, we used the concept of the *nodes* and also added some additional phase shifters to some of the whiskers that feed into the phase-sensitive modulators. *Full nodes* are indicated by the transparent blue overlay and their corresponding numbers (e.g. 2_0, 3_1, ...). These nodes, placed within the PIC cells, are connected by rib waveguides to form the full mesh. As shown in Fig. 3.16, we removed the phase shifters from some of the connecting waveguides, while in some of the whiskers (0_2, 0_4, 3_-1, 4_0, 4_6, 3_7) we added extra phase shifters, which creates a *custom node* in the mesh. Custom nodes A have the phase shifters positioned after the whiskers and are connected to the modulators. In contrast, custom nodes B have the phase shifters positioned before the whiskers and are connected to the monitor

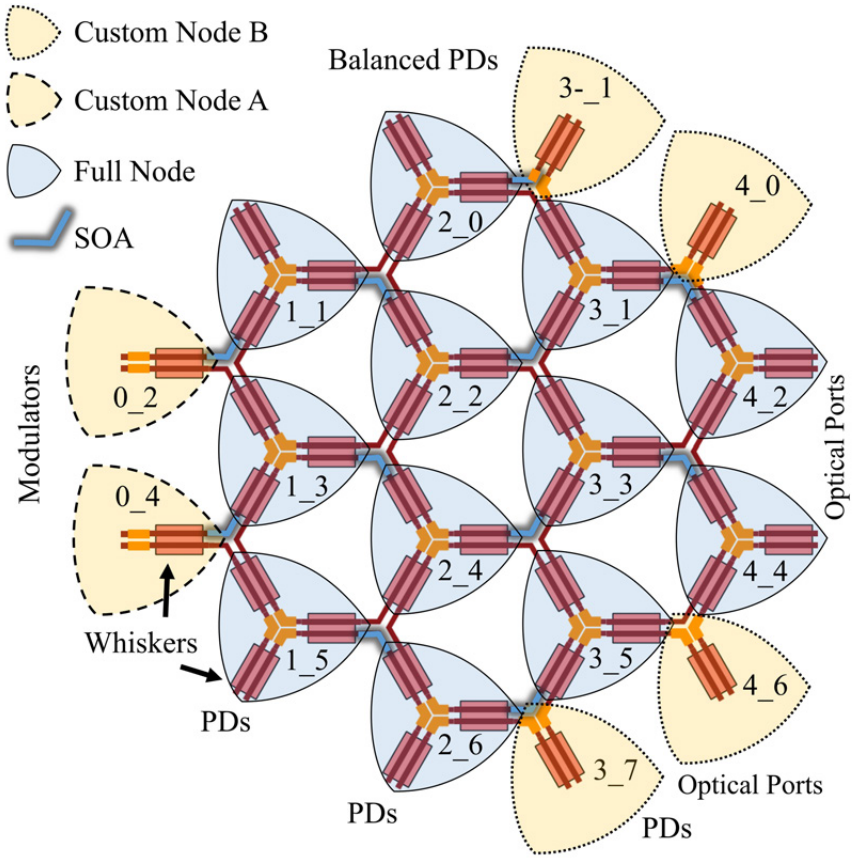


Figure 3.16: (Schematic of the mesh of the heater-based FP-PIC circuit designed for MORPHIC RUN2. This mesh is based on the 7-cell mesh with whiskers, and extra phase shifters have been added to make the mesh more practical, also the locations of the SOAs are indicated.

photodiodes, balanced photodiodes, and fiber array.

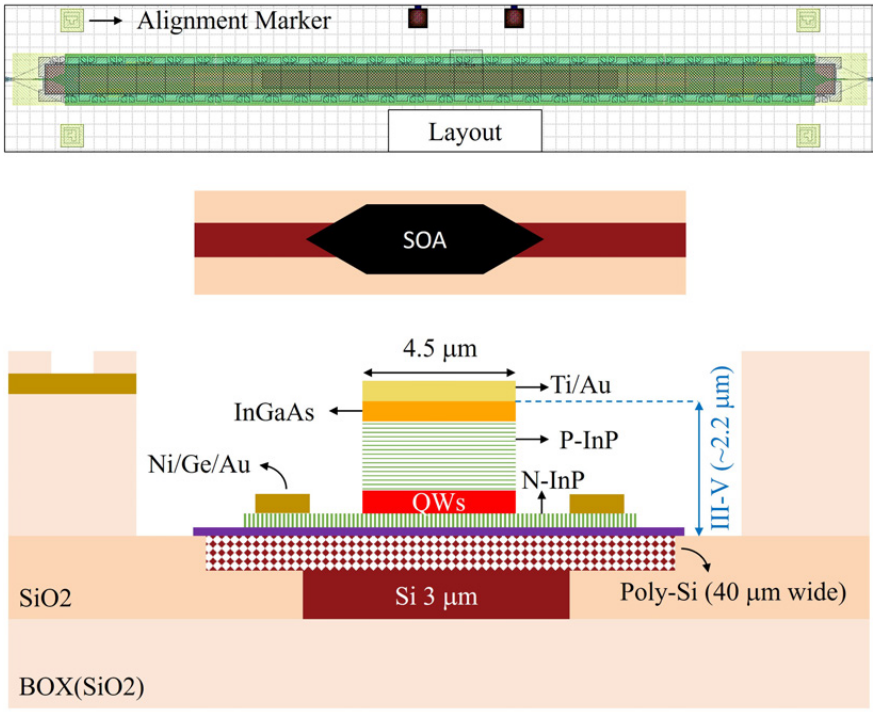


Figure 3.17: Schematic and layout of the Semiconductor Optical Amplifier (SOA) used for heater-based FP-PIC circuit on MORPHIC RUN2 [8, 9].

We used the opportunity to add transfer-printed SOAs to this heater-based FP-PIC to provide amplification of the light within the circuit as a compensation of optical losses caused by the its components. On the side of the silicon wafer, the SOA is a standard design which is processed as an etched cavity in the Back End of the Line (BEOL), down to a few hundred nanometers above the waveguide core. This is a similar process as the one used for exposing of the MEMS waveguides for the vapour HF processing. The SOAs themselves will be added through a micro-transfer printing technique [8, 9]. For the integration of the SOAs into our circuits we rely on component designs and fabrication processes developed by Dr. Jing Zhang and dr. Emadreza Soltanian. The transfer printing technology is being developed into a platform technology that is made accessible for circuit experiments like used here in MORPHIC. Figure 3.17 shows the schematic design of the SOAs. Possible challenges to implement this structure are etching back toward the backend stack to reach the Si layer, filling the side trenches with BCB polymer, and imperfect metal contact connections because of topographic difference of the pads.

Based on the available spaces in the circuit, size of the rims, and arrangement of their electrical pads, it was not possible to connect all ports of the nodes to the SOAs. Consequently, only one SOA per node was included. SOAs were arranged in a uniform way throughout the circuit by adding one SOA in each hexagonal mesh cell (Fig. 3.16). Note that the heaters and SOA do not require any sealing, but lids need to be added to provide a certain uniformity over the entire chip. Moreover, the hexagonal mesh has to be fitted in a rectangular grid of PIC cells compatible with the interposer used for electrically packaging the chip. Hence, each node, as indicated in Fig. 3.18, should be placed in one PIC cell.

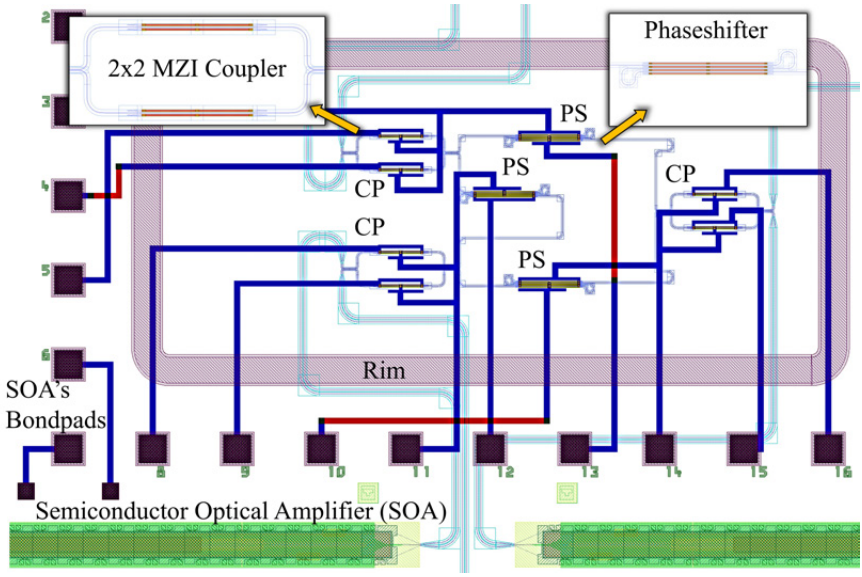


Figure 3.18: Layout of a PIC cell in the heater-based FP-PIC designed for the MORPHIC RUN2. The cell includes a full-node.

Figure 3.19 shows the final layout on the RUN2 mask, where the circuit consists of pre-tested phase shifters, tunable couplers (balanced MZI), and standard modulators and detectors from IMEC's iSiPP50G PDK. The two modulators are positioned on the West of the circuit, the BPDs are in the North, and fiber ports are on the East and South sides. All the other ports are connected to regular monitor PDs. SOAs are placed horizontally between the dummy lids (for process uniformity). As indicated, cell 0_4 is in the vicinity of the modulators which made it not possible to add a lid on this cell; similarly, the high density of the heaters in cell 3_1 also made it impossible to add a lid.

As seen, "node 3_-1" is placed near "node 3_1", this change helps to eliminated the extra cell on the North side of the circuit which leads to a more compact circuit

and improves floor planning. Also, two cells (3_-1 and 0_4) do not have lids. Here, 2 modulators, 4 pairs of balanced high-speed photodetectors, 4 low-speed monitor photodiodes, and 8 fiber ports are assigned. It is worth mentioning that the introduction of the SOAs breaks the symmetry of the 3 waveguides connecting each node. In fact, the segment with an SOA will have a significantly longer length than the other two waveguides.

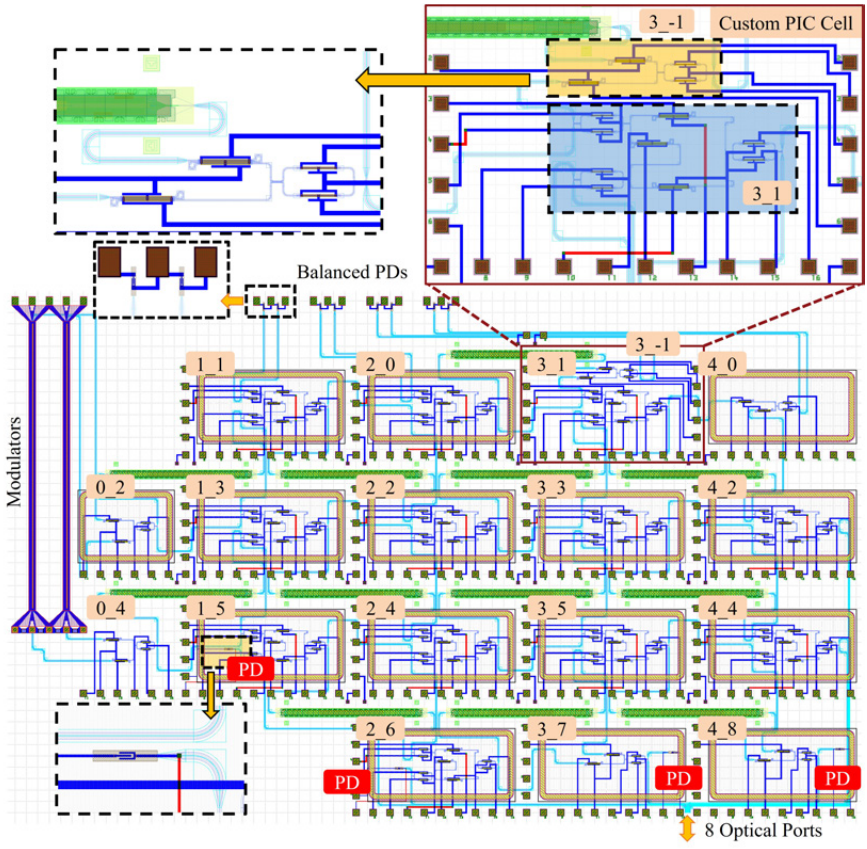


Figure 3.19: Final layout of the heater-based FP-PIC circuit designed for the MORPHIC RUN2.

To minimize thermal crosstalk of heaters and SOAs, the couplers and phase shifters are positioned more than $80\,\mu\mathrm{m}$ apart. Also, all the optical paths between couplers within a node are designed to be identical. The electrical wiring to the bondpads of the interposer unit cell are done in two-level metal, using wider wires for common ground lines of the heaters.

3.2.3 MEMS Building blocks

The implementation of MEMS technology on IMEC's iSiPP50G platform brings up a critical question: how do we connect the actuators to one another and to other circuit components? Specifically, we need effective interfaces to link the active MEMS devices to the standard iSiPP50G waveguides, as well as components to interconnect multiple MEMS devices within a single released cavity. As a result, in MORPHIC, we implemented a waveguiding scheme based on the fact that MEMS need to be released, or underetched, to enable the mechanical movement. In our approach, we have considered three regions as illustrated in Fig. 3.20a:

- **Encapsulated:** The waveguide is fully clad, with oxide layers both above and below it. This is the standard iSiPP50G configuration and is present in most areas of the chip where MEMS devices are not added.
- **Exposed:** The BEOL dielectric stack has been removed, leaving air above the waveguide and oxide beneath it. This occurs in a MEMS cavity (or in an area etched back to the waveguide surface) where the buried oxide layer (BOx) remains intact.
- **Suspended:** Both the top and bottom oxide claddings have been removed, typical of most MEMS devices, resulting in air (or vacuum) surrounding the waveguide core both above and below.

To transfer light between these regions we use strip, FC, and SKT waveguides as shown in Fig. 3.20a. SKT is a higher-contrast rib waveguide with $150\,\mathrm{nm}$ etch into the silicon while FC is a lower-contrast rib waveguide with $70\,\mathrm{nm}$ etch into the silicon; the default waveguide width of the SKT is $450\,\mathrm{nm}$, while for FC, is $650\,\mathrm{nm}$. The etching of grating fiber couplers is identical to that of FC waveguides, and the etching for electrical contact sockets (SKT) used in pn modulators follows the same procedure as for SKT waveguides.

As illustrated in Fig. 3.20b, the optical signal propagates through the MEMS cavity wall, passing through several cross-sections as it moves from an encapsulated to a suspended waveguide. This abrupt shift from silica-clad to air-clad waveguides can lead to reflections. To address this, FC- and SKT-based transition structures were designed in the MORPHIC project to minimize back-reflections and reduce insertion loss. Our measurements indicate that this approach achieves an insertion loss of $0.05\,\mathrm{dB}$ at $1.55\,\mu\mathrm{m}$ for both FC and SKT waveguide transitions. It is also worth to note that due to the nature of the processing flow, the exact positions of these intermediary interfaces are not precisely known.

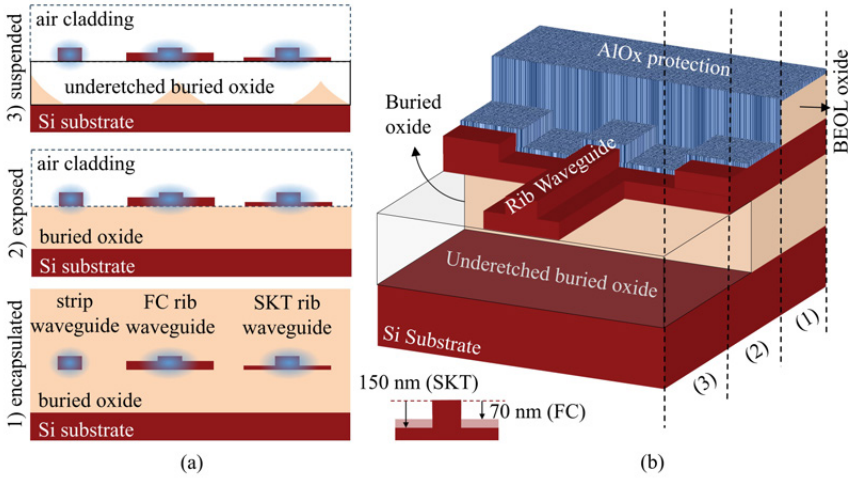


Figure 3.20: Schematic illustration of the strip, FC, and SKT waveguides designed to control light propagation across encapsulated, exposed, and suspended regions resulting from MEMS integration on IMEC's SiPP50G platform.

To have better understanding of the architecture of the MEMS blocks used in our circuit designs, we have shown two perspectives of a MEMS cavity in Fig. 3.21, where some of the material layers have not been shown to avoid unnecessary details for this discussion. As seen, the main components of the MEMS cavities are the cavity itself, the mechanical structure, the Si rim, electrical isolation trenches, optical transits, and electrical transits.

The alumina layer is for custom post processing and is added for protecting the Back-end layers against the aggressive HF(Hydrofluoric acid)/vHF(Vapour-phase HF) etching, and selective patterning of alumina provides electrical contact and access to MEMS cavities. The areas where the MEMS are designed utilize the EXPO etch module provided by IMEC to remove sections of the BEOL stack, exposing the device layer (DL) silicon and the BOX. Along with the patterned alumina, these openings define the MEMS cavities, as the alumina is removed in these regions, allowing the VHF to undercut structures.

Within each cavity, a silicon rim functions as an anchoring region and provides an additional protective barrier against VHF. Specifically, the rim prevents VHF from accessing the stack laterally, allowing entry only from below, which can only occur if the underlying BOX is entirely removed. It's important to note that a complete undercutting of the silicon rim is avoided, as the etching time is carefully controlled to suspend structures without causing this effect. Additionally, the rim houses the electrical isolation trenches that separate regions with different voltages

and supports the optical transitions necessary for low-loss passage between oxide- and air-cladding.

As seen in Fig. 3.21 the available design regions are separated by the MEMS rim. In fact, the MEMS rim region is enforced during the design phase and has been defined based on fabrication steps and post processing plans. The internal design area is related to the MEMS design and the outer one is for circuit design and high-level system considerations. For a detailed discussion of fabrication process of the MEMS cavities, you can refer to the MORPHIC publications, particularly the works of our colleagues at EPFL and KTH.

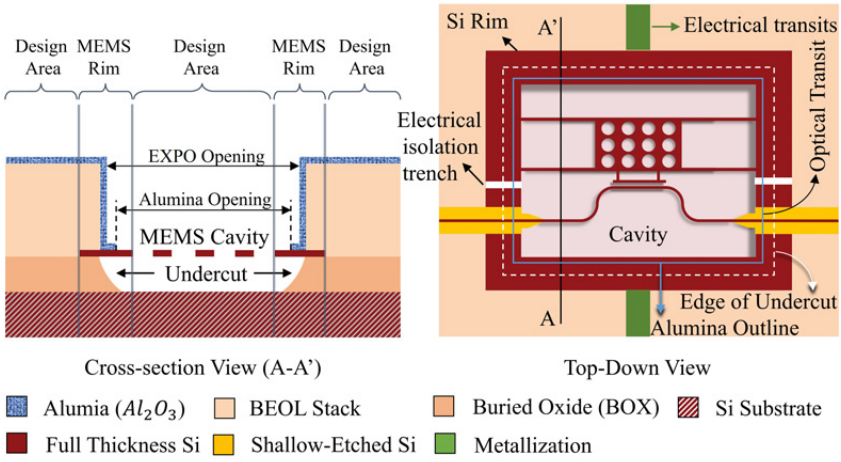


Figure 3.21: Schematic representation of a MEMS cavity containing a basic MEMS structure, shown from top-down and cross-sectional perspectives. Key features are optical transitions for optical I/O, electrical transitions provided by metallization, electrical trenches separating regions at different voltages, and Si rim. The MEMS rim divides design areas.

In large-scale circuits we have to use hundreds of MEMS structures which magnifies the effect of transition losses on the circuit performance. The ideal solution for this problem is to place all the MEMS devices in a super large cavity. However, due to fabrication and packaging limitations this approach is not practical. Hence, we have to carefully define reasonably large cavities to accommodate some of the MEMS components of each circuit. This results in defining a new waveguide connectivity. *Suspended Rib Waveguides (SRW)* which are not standard components in the iSiPP50G are responsible to connect MEMS actuator within a defined cavities.

Figure 3.22 demonstrates two connectivity schemes for the MEMS cavities: MEMS cavity-to-cavity connectivity (Fig. 3.22a) and MEMS inter-cavity connectiv-

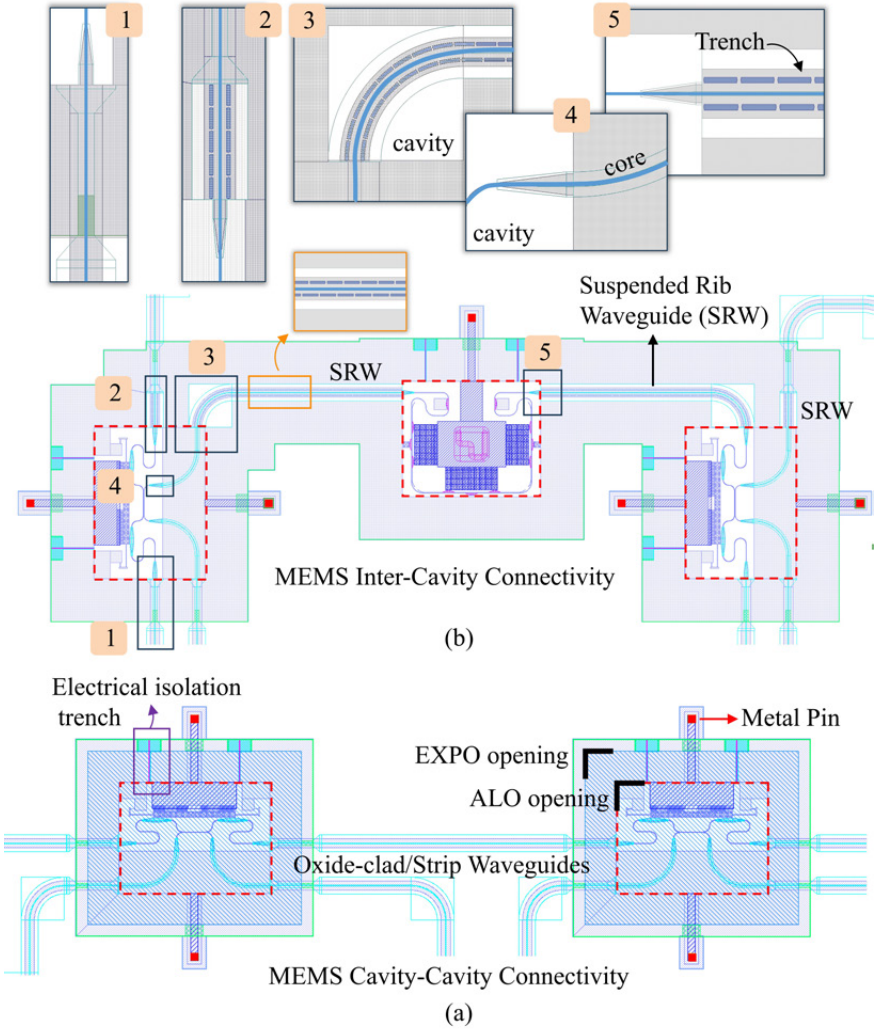


Figure 3.22: MEMS cavities connectivity schemes: a) Cavity-to-cavity and b) inter-cavity. In order to optically connect the structures we need optical transitions as indicated by 1-5 numbers.

ity (Fig. 3.22b) where the MEMS structure can be connected by the suspended rib waveguides. In the cavity-to-cavity scheme, each MEMS block can be individually wrapped in a 'rim' that allows for optical and electrical transitions through the MEMS cavity wall. These elements can then be treated as standard circuit blocks in iSiPP50G and interconnected using standard oxide-clad rib or strip waveguides. In the inter-cavity connectivity scheme, MEMS blocks can be connected within

a single cavity, which is then enclosed by a rim. This approach results in a more compact circuit and minimizes optical losses and back-reflections typically caused by transitions from air-clad to oxide-clad waveguides. However, this technique comes with certain limitations, as all electrical and optical ports of the MEMS sub-circuit must be accessible from the outside; routing metal wires through air-clad cavities is not feasible.

3.2.3.1 Actuators

As elaborated in Chapter 1, there are two actuation mechanisms for MEMS devices in the iSiPP50G platform:

- **In-plane actuation:** Here, the suspended waveguides are moved horizontally by electrostatic forces between two in-plane structures (e.g., a comb drive). This mechanism requires at least two electrical contacts to apply a voltage difference.
- **Vertical actuation:** In this mechanism, suspended structures are displaced vertically by a voltage difference between the free-standing structure and the silicon substrate. It requires one electrical contact on the top layer and a grounded substrate.

In our FP-PIC and switch circuits, we utilized these mechanisms to design and implement MEMS couplers and phase shifters, as illustrated in Fig.3.23 [1].

Coupler R2B: In our initial attempt during MORPHIC RUN2, we employed the coupler R2B, with a footprint of $251\,\mu\mathrm{m}\times251\,\mu\mathrm{m}$, for the crossbar 4×4 switch circuit (section3.2.7.1). This coupler relies on vertical actuation and includes an additional *dump* section that can absorb the input light when the coupler arms are in non-bar states. The detailed routing of light within this coupler is illustrated in Fig. 3.23a. This coupler was designed with a $150\,\mathrm{nm}$ gap, and therefore is prone to bridging, i.e. small silicon or alumina fragments in the gap of the directional coupler that mechanically connect the two arms of the coupler together.

Coupler R2A: This actuator was used for the 24-cell FP-PIC circuit (section 3.2.4) and the rest of switch circuits (they do not have any phase shifters). It operates based on the in-plane displacement, and to change coupling values of the coupler we use comb-drive actuation to change the gap between suspended waveguides in the evanescent coupling region. In this structure one arm of the evanescent coupler is fixed while another arm is connected to the comb drive facilitating the

[1] Couplers and phase shifters were designed by Dr. Hamed Sattari (EPFL) and Dr. Pierre Edinger (KTH)

mechanical movement. Note that reducing the gap size increases Δn_{eff} of the coupled waveguides; hence, increasing Δn_{eff} will increase the coupling which can reach to 1.0 as its maximum value. The footprint of the coupler is $171\,\mu\mathrm{m}\times206\,\mu\mathrm{m}$ with operating wavelength of $1.55\,\mu\mathrm{m}$; it has an extinction ratio of $25\,\mathrm{dB}$ and shows a broadband ($>35\,\mathrm{nm}$) optical power coupling. The initial waveguides gap in this structure is set to be $150\,\mu\mathrm{m}$ for the $0\,\mathrm{V}$ actuation.

Phase shifter R2A: This phase shifter was used for the 24-cell FP-PIC circuit (section 3.2.4). It has a vertical actuation mechanism and its footprint is $175\,\mu\mathrm{m}\times195\,\mu\mathrm{m}$. The initial waveguides gap in this structure is set to be $150\,\mu\mathrm{m}$ for the $0\,\mathrm{V}$ actuation. The phase shifter R2A consists of three vertically movable membranes that can tune the effective index of a $200\,\mu\mathrm{m}$ long waveguide surrounding the actuator (Fig 3.23c).

As shown in Fig. 3.23, we have major improvements for **coupler R3A** and **phase shifter R3A** used for MORPHIC RUN3 circuits: the footprint of MEMS actuators has considerably reduced and also the optical transitions have grounding. Their footprint is $84\,\mu\mathrm{m}\times131\,\mu\mathrm{m}$ for the phase shifter and $69\,\mu\mathrm{m}\times212\,\mu\mathrm{m}$ for the coupler.

Although both actuators operate based on the in-plane actuation mechanism, the phase shifter R3A is a dual-actuator component while the coupler R3A is a single-actuator. The designed phase shifter has a narrow silicon beam (loading structure) positioned close to an air-clad optical waveguide connecting the input and output of the device. The silicon beam, which is optically in cut-off mode at the operating wavelength of $150\,\mathrm{nm}$, acts as an index perturbation when brought close to the waveguide. Electrostatic forces are used to vary the distance between the optical waveguide and the silicon beam, thereby inducing a phase shift. When a voltage is applied across the comb drive (between the H-shaped shuttle and the fixed electrode), an electrostatic attractive force pulls the shuttle and, consequently, the silicon beam move away from the waveguide. On the other side, the springs provide a restoring force, causing the shuttle to return to its equilibrium state when the voltage is not applied. In fact, the phase shifter includes two opposing actuators, providing either gap-increasing or gap-decreasing displacements within a single device. It's important to note that reducing the gap results in larger phase shifts but also increases insertion loss. Conversely, increasing the gap leads to smaller phase shifts while keeping the losses low.

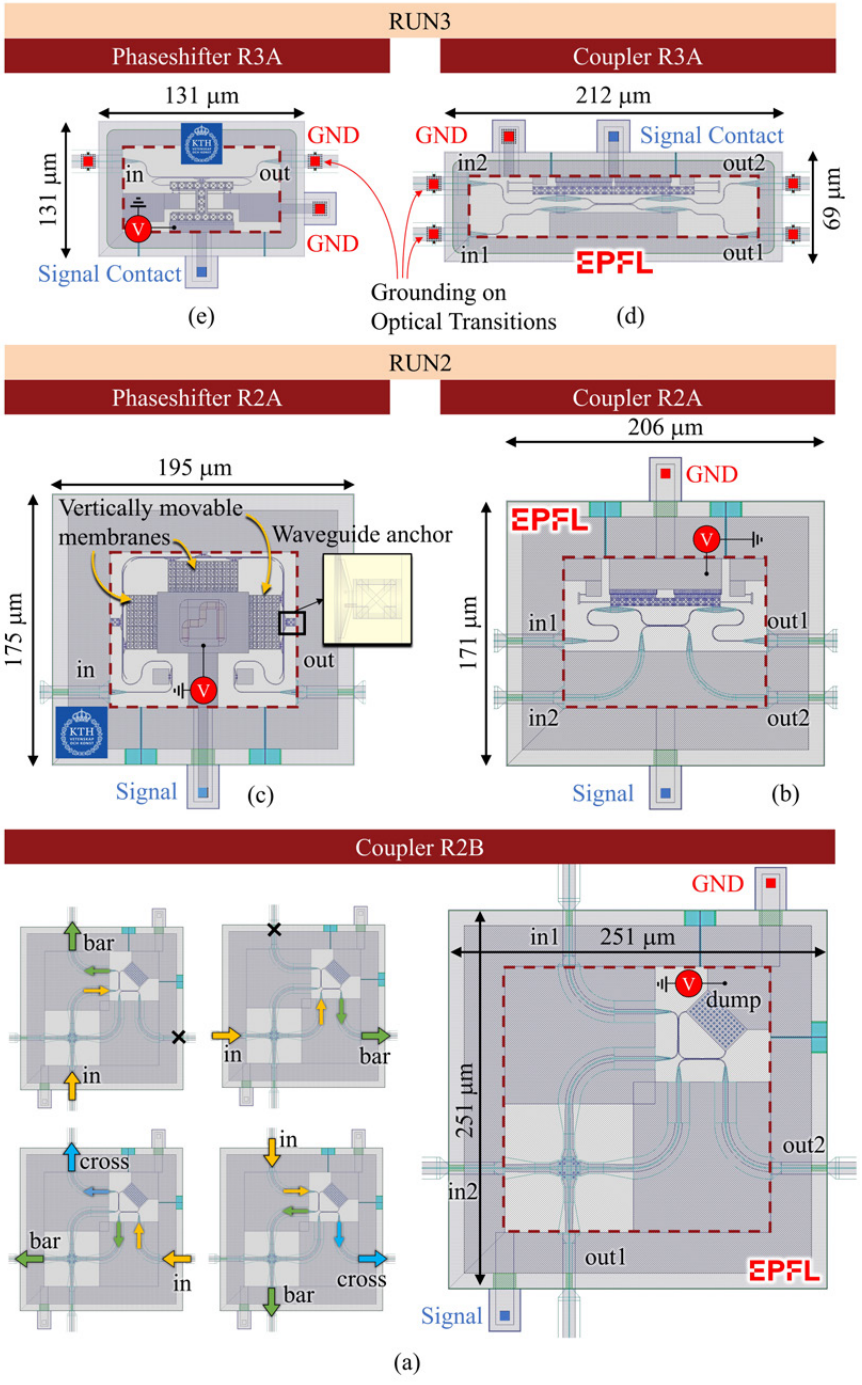


Figure 3.23: Summary of MEMS blocks used in circuit designs. They are wrapped in their own cavity (rectangles with dashed-line show the MEMS structures.)

3.2.3.2 Substrate Grounder

Another important component for designing our MEMS-based circuits is the *substrate grounding* (Fig. 3.24). Every PIC cell must include a substrate grounder. Substrate grounding is essential for out-of-plane actuators and strongly recommended for in-plane designs. The actuator consists of a flexible, single-clamped membrane that collapses with a low pull-in voltage of approximately 3 V. Once collapsed, the membrane adheres to the wafer substrate if the BOX layer has been fully etched away.

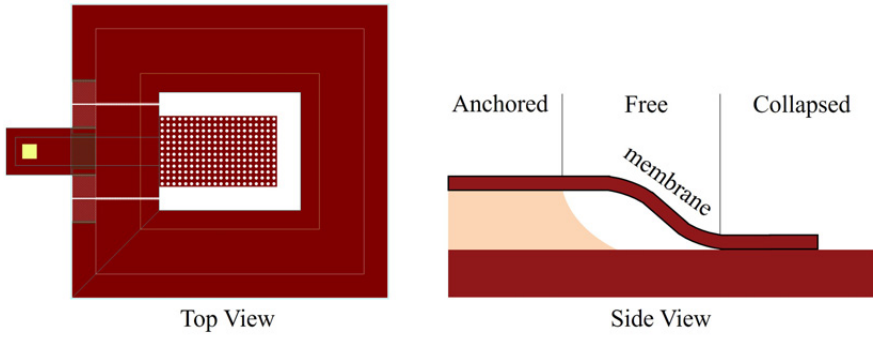


Figure 3.24: Simplified layout and side-view schematic of the substrate grounder used in the PIC cell for MEMS-based circuit designs.)

3.2.4 24-cell FP-PIC

For the 24-cell FP-PIC, we allocated the same area on the mask as the heater-based circuit. However, we decided not to include SOAs in this circuit. This made it possible to include two nodes into a single interposer unit cell, allowing us to add significantly more nodes in the mesh.

Figure 3.25 shows the topological schematic of the 24-cell hexagonal mesh used for the MEMS-based FP-PIC for MORPHIC RUN2. Compared to the 7-cell hexagonal mesh of the heater-based circuit, this circuit has an asymmetric shape. Due to the compact nature of the MEMS components and absence of SOAs, we were able to incorporate the entire circuit within a space similar to that of the 7-cell heater-based circuit. The 7-cell heater-based circuit occupies 17 PIC cells on the chip, while the 24-cell circuit occupies 19 PIC cells. The circuit includes 100 tunable couplers and 97 phase shifters, and, similar to the heater-based circuit, we have removed the phase shifters from half of the nodes. In Fig. 3.25, normal nodes (blue overlay), customized nodes (purple overlay), and the approximate location

and connectivity of the modulators, fiber ports balanced PDs and monitor PDs are indicated. The monitor PDs are spread around the circuit, the BPD ports are in the North, the modulators are connected to the North East, and the fiber ports are mostly in the East and South. To make the circuit more functional, some customizations have been applied. For instance, two extra phase shifters have been added to the cell 3_-1 in order to control phase shift of the waveguides connected to the modulators. We have connected two modulators, 8 BPDs, and 8 fiber ports, and 26 additional monitor PDs.

To design the circuit layout, we have considered two nodes per PIC unit cell as shown in Fig. 3.26. Each cell has 6 coupler R2A and 6 phase shifter R2A, unless there is a customization. Also, a substrate grounder and tabs with monitor grating couplers are dedicated for each cell. Since each node is positioned inside of a cavity the actuators are connected through the suspended rib waveguides for the inter-cavity connectivity as discussed before. And, we have dedicated two bondpads to ensure the grounding of the blocks.

Figure 3.27 shows the full layout of the circuit incorporated on the RUN2 mask. As mentioned, the two modulators are positioned on the North East of the circuit, the BPDs are in the North, and most of fiber ports are on the East and South side. All the remaining ports are connected to monitor PDs which are grouped in four locations on the South and East side of the circuit. One of the custom cells (0_4) is, also, highlighted in the figure; its right node is a full-node and the left node only

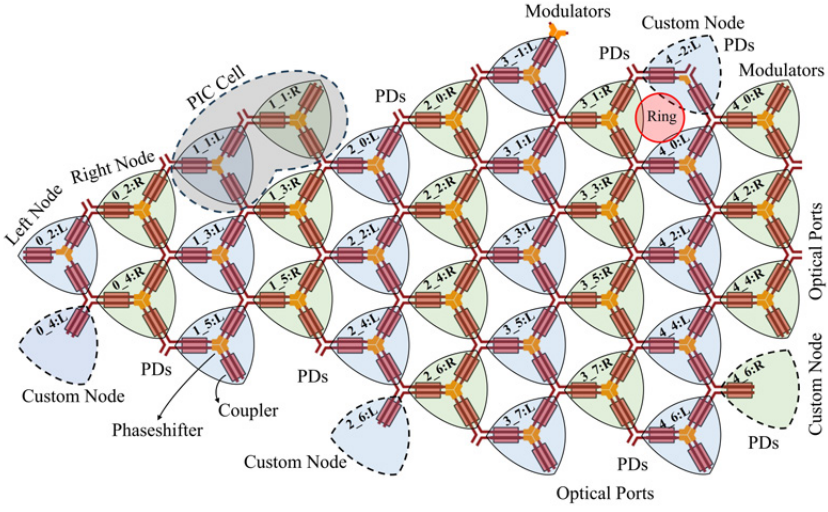


Figure 3.25: Schematic of the modified hexagonal mesh for the MEMS-based 24-cell FP-PIC in MORPHIC RUN2 including numbering system, normal nodes, and customized nodes.

contains a coupler connected to the two monitor photodiodes. This cell contains 6 monitor photodiodes grouped together.

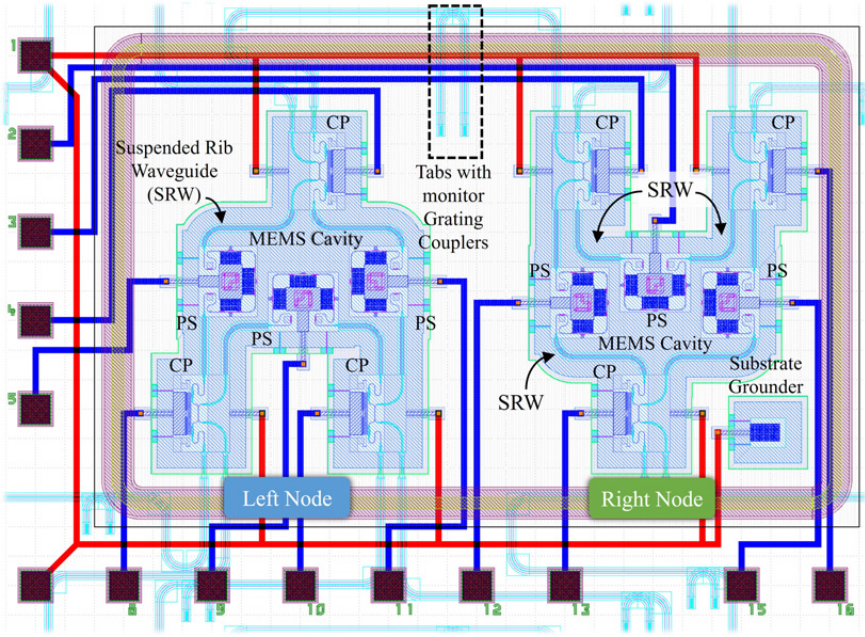


Figure 3.26: PIC unit cell containing 2 MEMS cavities consisting of 3 tunable couplers and 3 phase shifters, connected together with waveguides to the neighbouring cells. All phase shifters and couplers are connected to bondpads. Blue: metal1 (signal lines). Red: metal2 (ground lines). Also, all suspended rib waveguides (SRWs) between the MEMS cavities have a monitor tap connected to a grating coupler, for inspection with a camera.

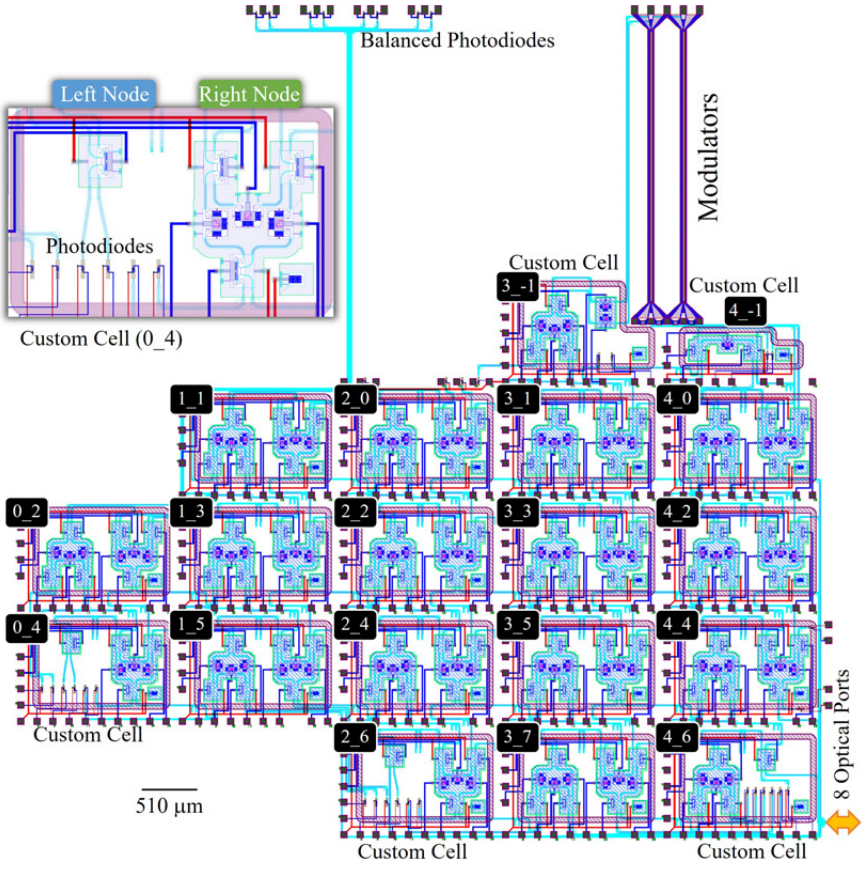


Figure 3.27: MEMS-based 24-cell FP-PIC circuit on RUN2.

3.2.5 126-cell FP-PIC

For MORPHIC RUN3, we designed a 9×14 hexagonal mesh in a parallelogram shape, consisting of 126 hexagonal mesh cells. Initially, we planned to use a rectangular mesh shape based on the analysis presented in the previous sections. However, we observed that when mapping the circuit mesh to the PIC layout, the rectangular shape transformed into a parallelogram shape on the chip grid. To account for this change, we designed the mesh in a parallelogram shape so that it would map as a rectangular shape on the chip grid. This approach allows other designs to be placed neatly alongside our circuit on the chip.

The circuit mesh is connected to 4 high-speed modulators, 8 balanced photodetectors and two SOAs, making it possible to demonstrate the different demonstrator functions. These active components have been located on the North side of the mesh as shown in Fig. 3.28. To construct the mesh we have used two types of the PIC cells as highlighted in Fig.3.28. 2-node PIC cell is composed of two full nodes, and 4-node PIC cell includes 6 custom nodes where each one has only 3 couplers. As a result, the mesh has two different sections:

- including phase shifters (blue cells, 2-node PIC cell).
- without phase shifters (red cells, 4-node PIC cell).

The reason of such division is to avoid unnecessary losses caused by the phase shifters and to reduce the overall footprint of the circuit. Sections without phase shifters are used for the routing and switching purposes, while center of the circuit can perform phase-sensitive functions such as wavelength filtering or a beamformer network.

The designed mesh has 16 fiber ports on the East (output) and the West (input) sides, and it is expected to operate as a 16×16 switch and other routing/redistribution functions. The South part of the mesh will act as a 16-channels beamformer. And a microwave photonics processor with phase and amplitude modulation, preferably over multiple channels, will be handled by the North and the center parts of the mesh, where the active and high-speed components are located.

The implementation of two types of nodes for the mesh topology results in two distinct types of circuit unit cells on the actual chip as mentioned and has been illustrated in Fig. 3.29. Each PIC unit cell has 16 pads, where one pad is for grounding, 3 pads are for monitor photodetectors, and 12 pads are for actuators. To ensure safe grounding everywhere in the circuit, all grounds are connect to each other in a large mesh on the Metal2 layer. Considering the number of bondpads available for each circuit unit cell and the footprint of the MEMS components, we

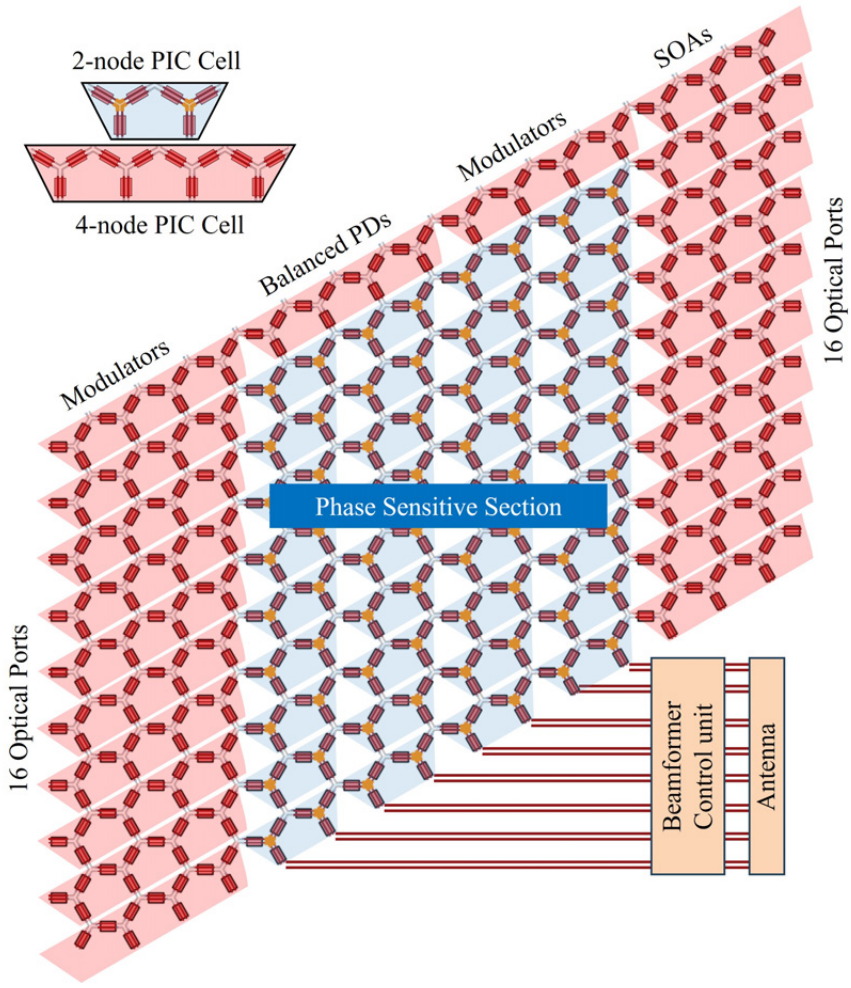


Figure 3.28: Overall architecture of the FP-PIC design on MORPHIC RUN3. We chose a parallelogram-shaped mesh with parts that are phase sensitive and parts that are not phase sensitive.

could position up to 12 MEMS components in each unit cell.

For path balancing, two sets of waveguides have been designed, each with equal length within a set. Waveguides inside each full node connecting two couplers (considering phase shifter optical length) have the same length and number of bends. Similarly, waveguides connecting the full nodes also have identical lengths. For the rest of the waveguides in the custom nodes we have used the shortest possible length along X and Y directions (so-called Manhattan-geometry): the main purpose

of these waveguides is connectivity of the cells, routing, and switching operations. Similar to the 24-cell FP-PIC, we have used suspended rib-waveguides to connect the MEMS structures inside of the nodes' cavity.

Figure 3.30 shows the layout of 126-cell FP-PIC mesh designed for MORPHIC RUN3. As seen high speed components and SOAs are placed on the north side of the circuit. We also have added two extra customized PIC cells (highlighted in the figure) to connect the mesh to the modulators.

As we discussed earlier, the FP-PIC circuits includes an ASPIC unit for beamforming (Fig.3.31a). In fact, one of the important demonstrators configurable in the mesh is a 1×16 beamformer. The outputs of the beamformer are located on the South side of the mesh (Fig.3.31a) and are coupled to a path-length-matching bundle of waveguides that are connected to the beamformer unit. However, due to the limitation of the space on the chip we had to put is far away from the circuit mesh (Fig. 3.31b). The beamformer unit consists of 16 independent phase shifters (Fig. 3.31d) and is connected to a 16-channel grating coupler antenna (Fig. 3.31c). We also have shown the position of the all monitor photodetectors in Fig. 3.31a, ans as seen, they have properly covered the area in which the beamformer can be configured. The 1×16 beamformer can be configured within the mesh in two different schemes: tap-based and standard tree. Figure 3.31a shows an example implementation of a standard tree in the mesh, the starting point is accessible by the fiber ports on West and East sides of the mesh.

One of the related functions of the beamformer not implemented in the FP-PIC

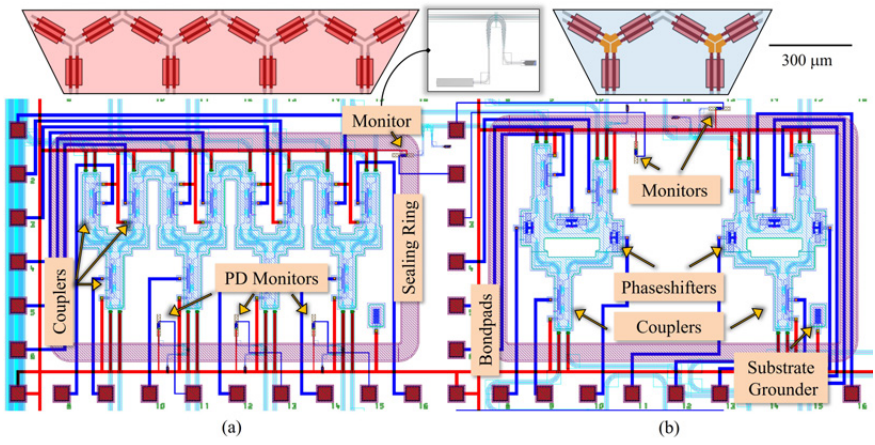


Figure 3.29: Mask layout of the PIC unit cells of the FP-PIC on RUN3. (a) a 4-node unit cell without phase control, (b) a 2-node unit cell with phase control.

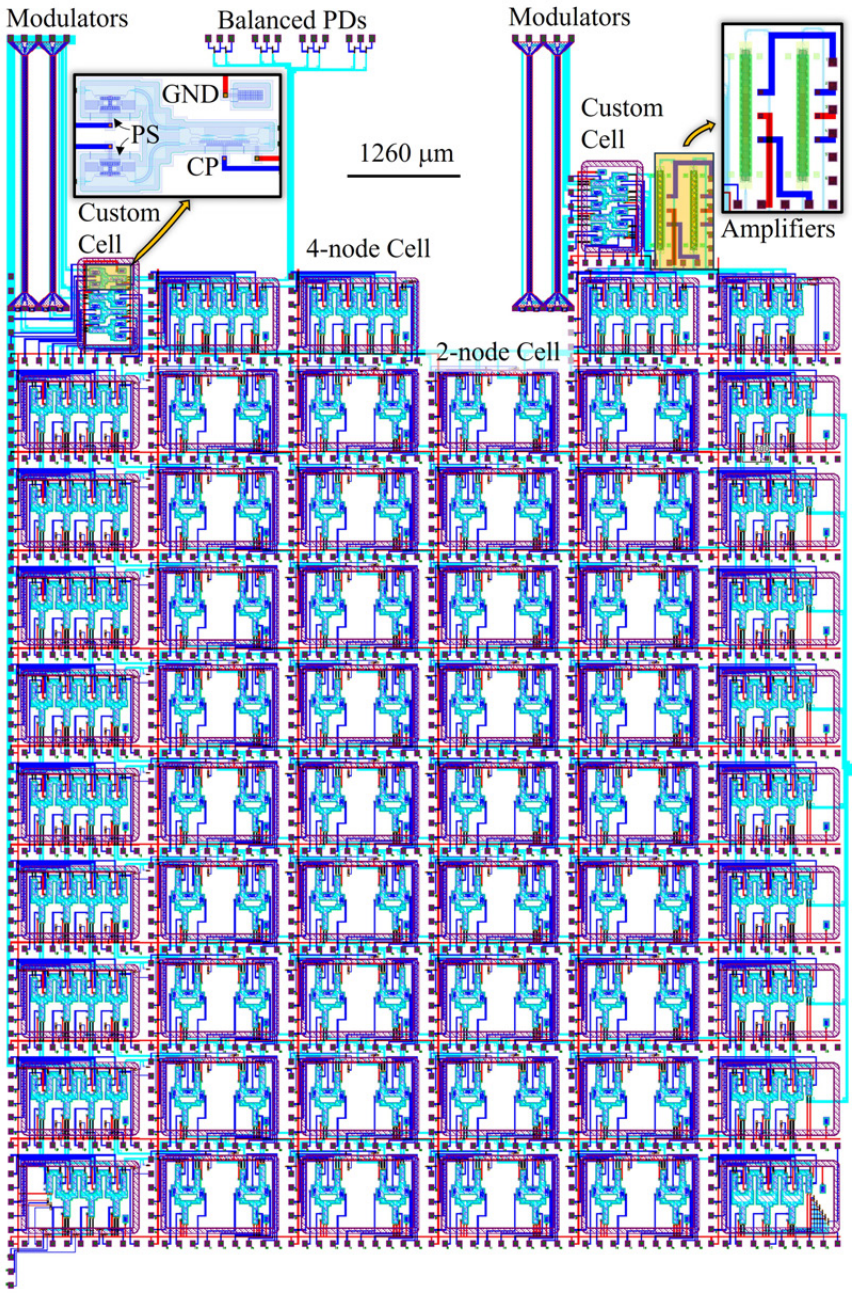


Figure 3.30: Mask layout of the 126-cell FP-PIC on RUN3.

is the monitoring of phase and amplitude at the output of the beamformer (after the phase shifters). This limitation was due to the unavailability of bond pads in that area of the chip. As a result, the beamformer control will have to rely on pre-calibration instead.

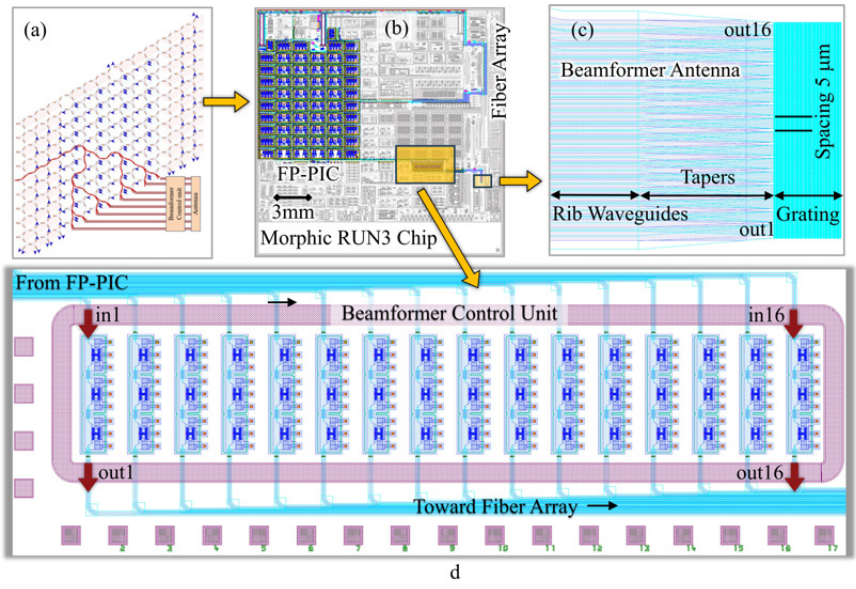


Figure 3.31: a) Schematic implementation of a 1×16 beamformer on the circuit mesh. The beamformer outputs are routed to the control unit, consist of phase shifters, using balanced waveguides. The control unit is connected to the grating couplers antennas on the other side. b) Mask layout of the 126-cell FP-PIC on RUN3, where location of the beamformer control unit and corresponding antenna array are highlighted. c) Beamformer antenna array and its rib waveguide connectors. d) The beamformer control unit consists of an array of 16 sets of 3 connected phase shifters, designated as R3A.

One of the advantages of the parallelogram-shaped mesh for the FP-PIC circuit is the efficient use of couplers in each unit cell, with 4 out of 6 couplers being actively utilized (excluding unused whiskers on the edges). The switching scheme is illustrated in Fig.3.32 for a switch matrix with 6 channels implemented in a 3×4 parallelogram-shaped mesh. As shown, for 8 channels, 3 complete rows of unit cells are required. For full, non-blocking switch functionality, 3 complete columns of cells are also necessary. A quick numerical scaling analysis indicates that this scales linearly with the number of ports. For $2N$ ports, N full rows and N full columns are needed, for $N > 2$. This means that for a 16×16 switch, at least an 8×8 mesh is required. Additionally, as seen in the schematic in Fig.3.32, tunable couplers or switches are needed at the input and output ports.

It is worth noting that the FP-PIC design, with its 9×14 cells, is overdimensioned. This extra capacity provides redundancy: if a switch becomes stuck in either the cross or bar state (but not in a partial state), the additional columns enable us to reroute the path. Since the switch matrix functionality does not depend on phase-sensitive operation, all unit cells in the mesh can be utilized.

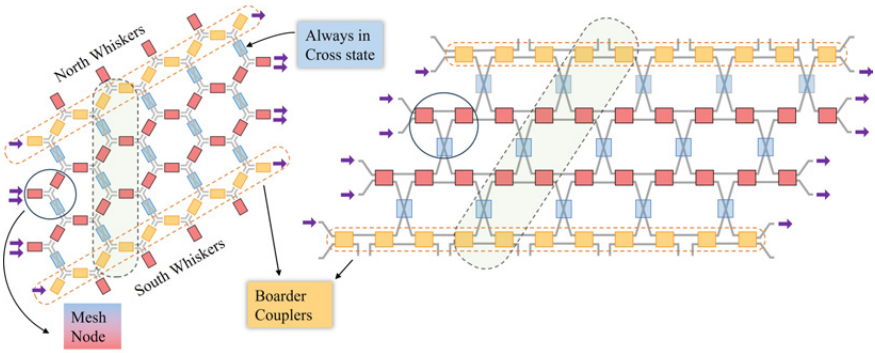


Figure 3.32: Switching architecture in a Parallelogram-shaped mesh of hexagonal unit cells. Left: Mesh configuration, using the red and yellow couplers for switching. The North and South Whiskers are not involved in switching process, the blue couplers should be always in cross state. Right, the equivalent switch topology.

Figure 3.33 illustrates an example of a microwave circuit featuring a 4-ring RAMZI (Ring Assisted Mach-Zehnder interferometer) configured within the FP-PIC mesh. RAMZI is a specific type of photonic circuit used in optical signal processing. It combines the functionalities of a Mach-Zehnder Interferometer (MZI) with ring resonators to enhance performance and enable more complex operations. The ring resonator enhances the MZI's ability to filter specific wavelengths, increase extinction ratios, and improve the finesse of the interferometer. The RAMZI can be used to achieve narrowband filtering, wavelength-selective switching, and even more advanced signal processing tasks that are critical in dense wavelength-division multiplexing (DWDM) systems and other high-performance optical networks.

As shown, the laser input from the fiber port splits into a reference path and a signal path; the signal path then passes through the modulator and subsequently through the filter. Afterward, the reference and signal paths are recombined in a 50:50 coupler before reaching a balanced PD. It is important to note that such configurations should be designed to minimize routing loss, which necessitates the use of efficient routing algorithms.

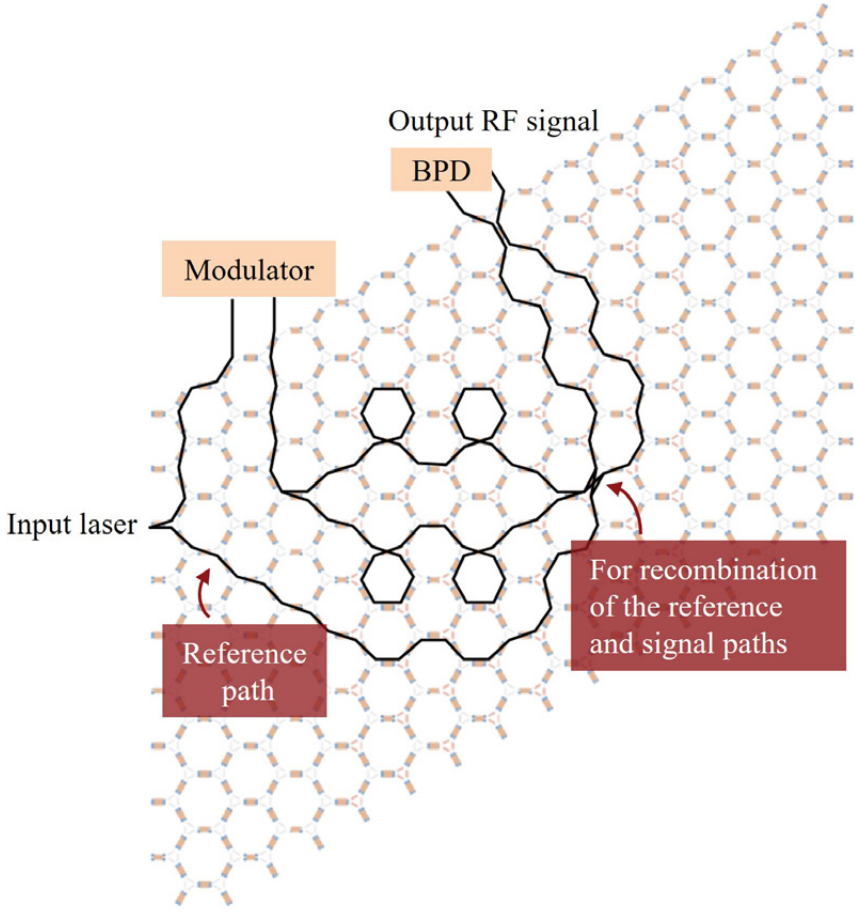


Figure 3.33: Schematic implementation of a 1×16 beamformer on the circuit mesh. The beamformer outputs are routed to the control unit, consist of phase shifters, using balanced waveguides. The control unit is connected to the grating couplers antennas on the other side.

3.2.6 Test Nodes

In addition to the 126-cell FP-PIC, we designed two test circuits using isolated full and custom nodes and placed them within a PIC cell, as shown in Fig. 6.47. Given the high rate of MEMS collapse in MEMS-based circuits, these designs were created to increase the likelihood of survival and to collect measurement data for developing a circuit model of these nodes. This model could then be used in circuit simulations to yield more realistic simulation results.

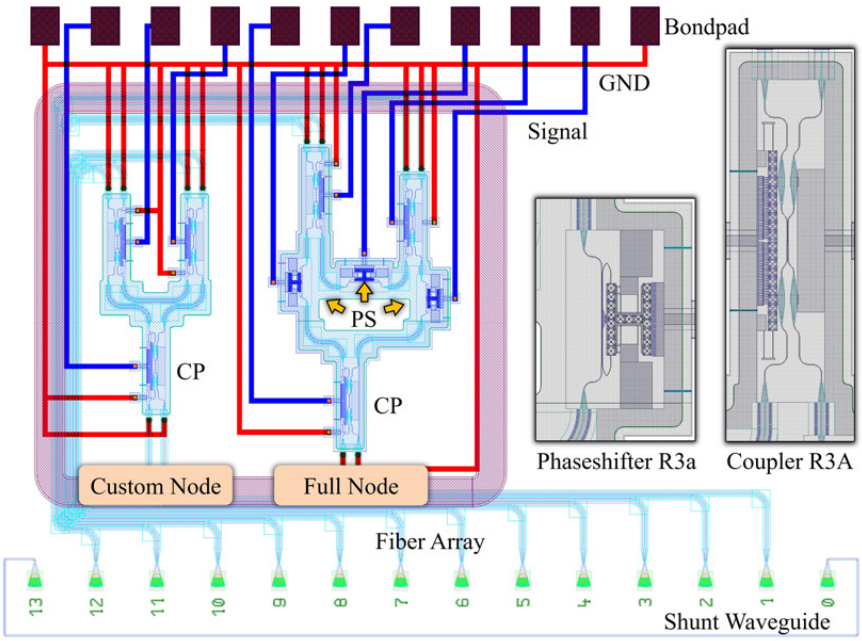


Figure 3.34: Mask layout of the test nodes on MORPHIC RUN3.

3.2.7 Switch Circuits

Optical switch matrices have already been successfully demonstrated across various PIC technologies [106]. As part of MORPHIC RUN2, we incorporated different types of switch circuits using the photonic MEMS technologies. We implemented 3 different architectures [2] in different sizes, to experimentally study the scalability and compare their performance with their twins implemented in FP-PICs' mesh. Designed switches are:

- **Benes:** This is the most compact architecture, requiring the fewest switches. However, it is not non-blocking, meaning that reconfiguring certain channels can affect others. Additionally, the losses through the network vary across different paths.
- **Path-independent loss (PI-loss):** This architecture is considerably larger but ensures a uniform path length for all channels, as the signal passes through the same number of crossings and switches, regardless of whether they are in the cross or bar state.

[2]The layouts were originally developed by B. Abasahl at IMEC.

- **Crossbar:** This is the simplest switch network, with a size comparable to the PI-loss architecture, but it does not offer path balancing.

For the Benes and PILOSS circuits a simple 2×2 directional coupler (coupler R2A in Fig.3.23) is utilized. This coupler is versatile, allowing for any coupling ratio between 0-100%. For the crossbar network, a combination of two unidirectional 1×2 switches can be employed (coupler R2B in Fig.3.23). This switch offers the advantage of low crosstalk and a high extinction ratio compared to the simple coupler, but it is limited to 0% or 100% coupling, without intermediate values.

The switch circuits discussed in this section, which will later be used for our measurements, include the crossbar 4×4, Pi-loss 4×4, Benes 4×4, and Benes 16×16. It is worth noting that other switches with different dimensions were implemented in MORPHIC; however, this work focuses only on these specific switches.

3.2.7.1 Crossbar Switch

Figure 3.35 shows the layout and schematic of the Crossbar 4×4 switch circuit designed for MORPHIC RUN2. This is the most straightforward network to configure, but it is also the one with the largest number of switches. It contains 16 MEMS couplers, 4 inputs, and 4 outputs where the inputs are at the North side of circuit and outputs are on the East side. The advantage of coupler R2B (Fig. 3.23a) used for this switch is that light passes through two couplers, which suppresses the crosstalk. However, it is not possible to operate it as a fractional coupler, making it impossible to use this element in a multicasting network. In this circuit, the dead ends on each switch elements need to be optically terminated to avoid spurious back-reflections that can cause cavity interferences. Since the crossbar matrix is non-blocking, a change in the connectivity will only affect the signals that are being altered, and the rest of the input-output combinations will remain the same.

3.2.7.2 PI-loss Switch

The path-independent loss architecture, as its name implies, provides a network topology where all optical paths have identical properties in terms of waveguide length and switch states, regardless of the network configuration. Specifically, in this architecture, all paths have their switches in the ‘cross’ state, except for one, resulting in uniform transmission properties across all channels. The PI-loss architecture is not only highly uniform but also non-blocking. When the matrix is reconfigured for a new connection, only the connections being adjusted are affected,

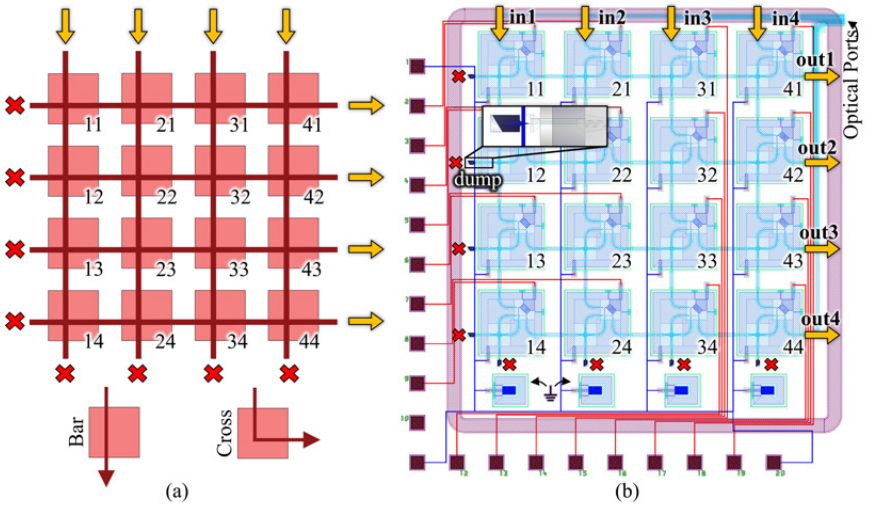


Figure 3.35: Crossbar 4×4 switch circuit. a) schematic b) layout.

while the switch matrix remains transparent to other connections.

Figure 3.36 shows the schematic and layout of the PI-loss 4×4 designed for MORPHIC RUN2. It contains 4 inputs (West side) and 4 outputs (East side), and the number of the used couplers is 16, like with the crossbar switch. This circuit also has 4 dump ports, but unlike the Crossbar 4×4, it allows multi-path interference.

3.2.7.3 Benes Switch

The simplest switch designed in MORPHIC RUN2 is the Benes 4×4 switch circuit (Fig. 3.37). It has 6 couplers, 4 inputs (left side) and 4 outputs (right side). In contrast to the Crossbar 4×4 and the PI-loss 4×4, this circuit does not have dead-end beam dumps. In other words, all light injected in the circuit should reach the end, except for propagation losses and reflections along the way.

The Benes network, named after Václav E. Beneš, is a rearrangeable non-blocking network architecture. When two sets of input-output connections are changed, at least one (and often several) additional switches need to be adjusted, causing temporary disruption to the traffic through part of the network during the switching process. Consequently, this architecture may not be suitable for applications where even brief signal interruptions cannot be tolerated. The Benes switches exhibit low but variable transmission loss due to the differing path lengths within the network, which result in varying numbers of crossings and switches in

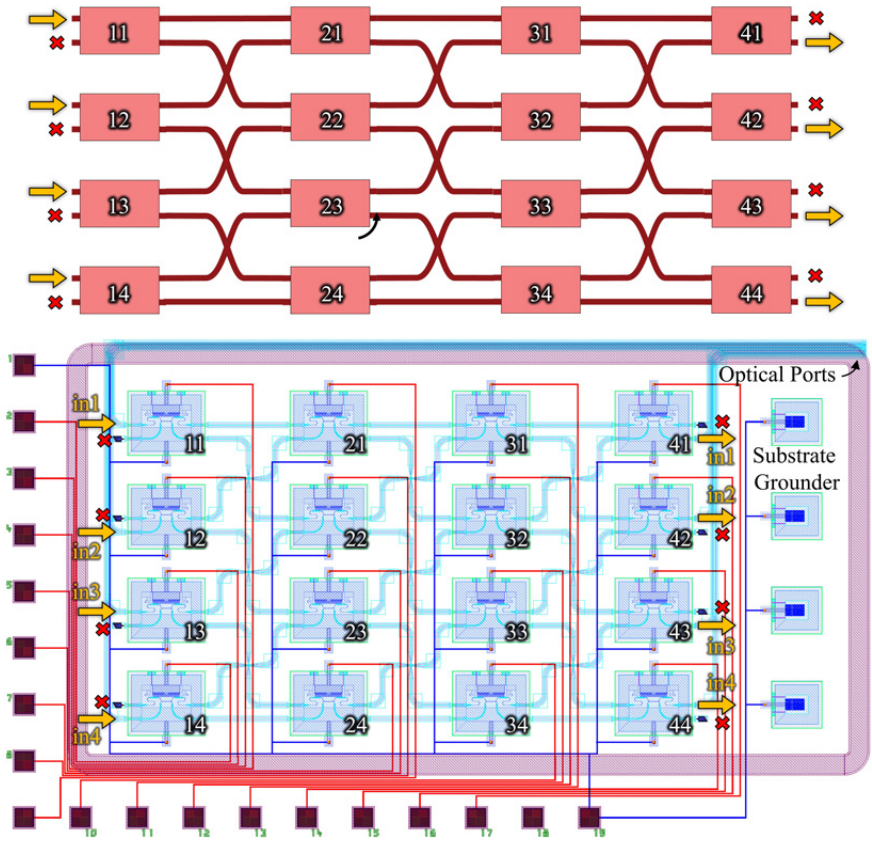


Figure 3.36: PI-loss 4×4 switch circuit.

either the 'bar' or 'cross' state along each path.

The Benes switch network can connect 2^{LV+1} ports to any of the outputs, where LV represents the circuit's level. Each network consists of two sub-circuits at $level = LV - 1$, surrounded by arrays of $N/2$ switches. The Benes 4×4 switch has a level of 1, connecting 4 input ports to 4 output ports. Additionally, we designed a larger Benes switch (Benes 16×16) with a level of 3 to test the scalability of this switch type. Figure 3.38 shows the layout of this switch, with the Benes switches at level 1 and level 2 highlighted. Both of the designed Benes switches feature a single substrate grounder, unlike the other implemented switches.

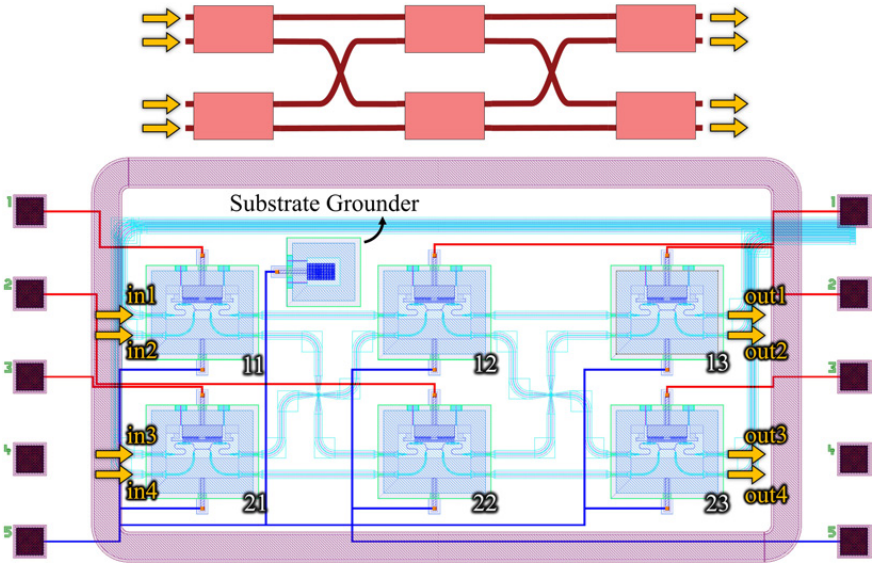


Figure 3.37: Benes 4×4 switch circuit layout.

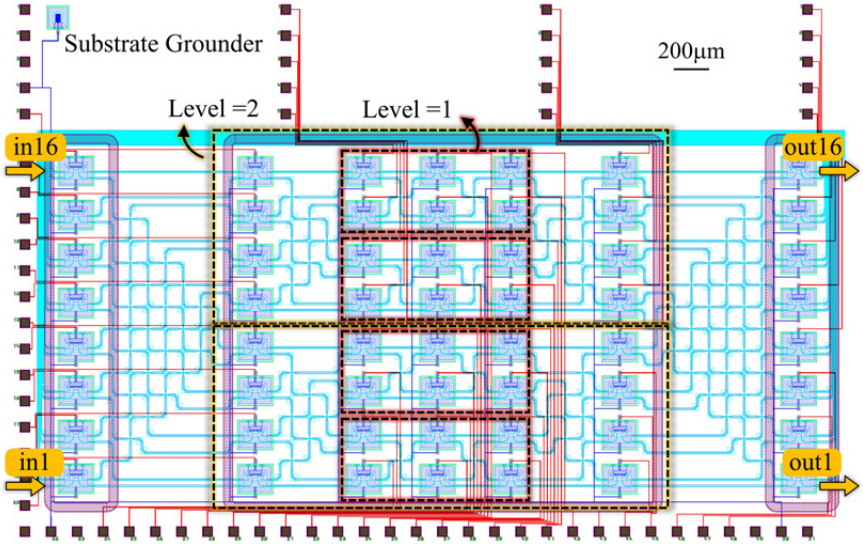


Figure 3.38: Benes 16×16 switch circuit layout.

3.3 Conclusion

In this chapter, we presented both loss and scaling analyses for hexagonal meshes of various shapes. Our findings indicate that rectangular meshes are the optimal choice for designing FP-PIC circuits. However, since implementing a rectangular-shaped mesh on the actual chip could disrupt the organization of other circuits, we opted for a 126-cell parallelogram-shaped mesh for the MORPHIC RUN3 design. Additionally, two smaller circuits—one based on heaters (a 7-cell FP-PIC) and the other on MEMS (a 24-cell design)—were created for MORPHIC RUN2. The goal of the FP-PIC designs in RUN2 was not only to test the developed MEMS platform but also to compare the circuit performance of the two different approaches using MEMS and heater actuators.

Additionally, the principles behind constructing the MEMS cavities have been explained, along with the various optical transitions used to connect the MEMS actuators to external components. The concept of PIC cells for implementing the mesh nodes is also covered. Each PIC can accommodate up to 12 actuators, with the primary limiting factors being the size of the PIC cell rim and the number of available electrical bond pads. All FP-PIC circuits benefit from active components such as modulators and balanced photodiodes; however, the 126-cell FP-PIC and the 7-cell heater-based FP-PIC also incorporate SOAs. The 126-cell FP-PIC is further connected to a special unit, a 1×16 beamformer control unit, which is linked to a fiber grating array antenna. We also placed two isolated full and custom nodes as our test circuits in MORPHIC RUN3.

To enable a direct comparison between the Benes, Crossbar, and PI-loss architectures, we implemented a 4×4 switch matrix for each architecture, as well as a 16×16 switch matrix for the Benes. It is evident that the Benes architecture is the most compact of the three, while the Crossbar and PI-loss architectures use the same number of switch elements. The Crossbar is larger due to its use of a bigger building block with the double-coupler switch. Ideally, these circuits should be configurable within the FP-PIC circuits, providing valuable insights into the performance of FP-PICs compared to ASPIC circuits.

We should already mention here that of these many circuit designs, most could not be tested. As we will discuss further, these circuits are very large and need to go through a complex packaging process. For reasons that are not yet entirely resolved, many of the MEMS building blocks fail during the packaging process, which made the circuits non-operational. As we discuss in Chapter 5, we did perform a thorough characterization of several circuits to analyse the extent of these failures.

4

System Architecture and Integration

In this chapter, we present the overall structure of the electro-optic system designed to operate MEMS-based reconfigurable circuits. We focus on the system's modular design and discuss its architecture from both a hardware and software perspective. During the MORPHIC project, we had the chance to fabricate our reconfigurable circuits for two of IMEC's dedicated runs: RUN2 and RUN3; we detailed the circuits implemented for each run in the previous chapter.

One crucial step in preparing photonic chips for packaging involved sealing the MEMS components, a process conducted by KTH, which is briefly covered here. We also examine the packaging processes and strategies employed by Tyndall throughout the project. We developed two packaging approaches, referred to as full and mini demonstrators. The full demonstrators, featuring complex PCBs and interposers, allow for control over all fabricated circuits on the RUN2 and RUN3 chips. Conversely, mini demonstrators target specific circuits and utilize simpler interposer and PCB designs.

We provide a detailed overview of the electronic hardware (developed by Tyndall) essential for controlling and interfacing with the photonic integrated circuits (PICs), primarily focusing on the Electronic Interface and Control (EIC) boards and the BeagleBone—a single-board computer. The EIC boards, equipped with high-voltage drivers and photocurrent readout circuits, are vital for managing the PICs. The BeagleBones act as controllers for the EIC boards, executing local

control tasks directly on the EIC boards and centralizing more complex control operations, showcasing a modular electronics architecture that enhances scalability and flexibility.

Finally, we introduce the software control layers designed to manage and optimize the programmable photonic circuits developed with integrated MEMS technology. This software stack is essential for facilitating dynamic, reconfigurable control over the photonic components and their functionalities within the Field-Programmable Photonic Integrated Circuit (FP-PIC) platform.

It is important to note that the complete system, comprising photonic chips, electronics, packaging and software, is the result of collaborations in MORPHIC by multiple people from different project partners. Also, given that some technical details are still considered confidential, I will provide a general overview here, with an emphasis on my own contributions. My primary contribution includes photonic chip design and characterization, interposer and PCB interconnect design, software development for the system control, and system test by using electronic drivers to reconfigure the circuits on the chip and consequently measuring optical responses.

4.1 Introduction

Figure 4.1 illustrates the schematic of a programmable photonic system architecture. This system comprises a photonic chip, an interposer, a PCB interconnect, electronic drivers and controller (EIC boards and BeagleBones), optical switch, RF/optical sources and monitors, and user PC to control the system. Consequently, assembling such a system necessitates multiple interfaces, which are influenced by the actuator technology used in circuit fabrication, the number of electrical connections and optical ports on the photonic chip, and the RF specifications. The interfaces identified in our system design are:

- **On-chip interfaces** include both electrical and optical interfaces. For the electrical interface, a bondpad grid (section 4.2.4) is used, where each bondpad is constructed from metal layers on IMEC's iSiPP50g platform (section 1.5). And, the optical I/O of the photonic chip is accommodated by two sets of 72-fiber arrays, as will be discussed later.
- **Packaging interfaces:** To interface the PIC components (actuators, monitors, and modulators) with the electronic control and optical ports to the laser source and detectors, a dedicated packaging approach should be setup. The photonic circuits, especially the large-scale programmable PICs require hundreds of electrical contacts that need to be addressed simultaneously.

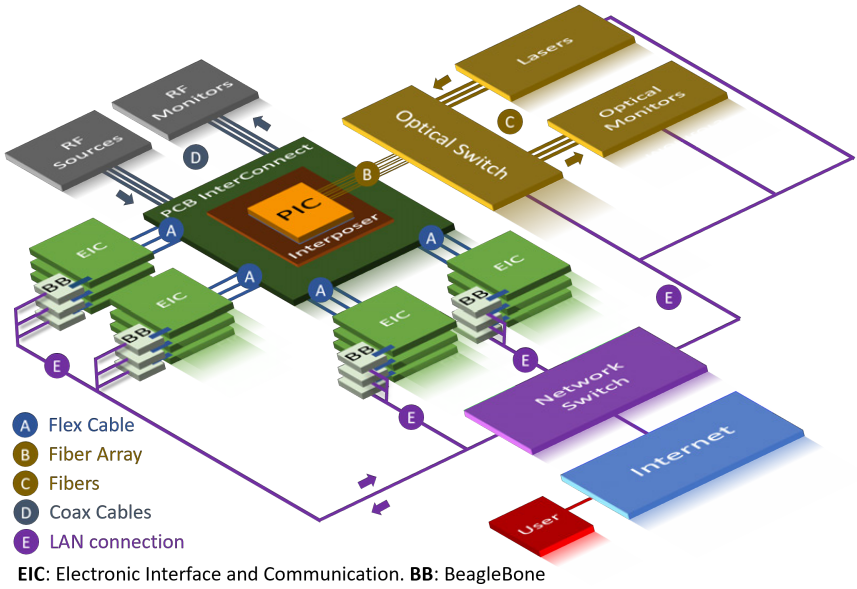


Figure 4.1: Schematic of a programmable photonic system architecture.

In addition, many optical interfaces, as well as high-speed electrical (RF) interfaces are needed. Since the circuits also include high-speed photodetectors (Balanced Photodetectors (BPDs)), suitable trans-impedance amplifiers should be integrated near them.

For this means we are using both high-density and low-density interposers to manage electrical connections for the MEMS actuators and the monitor photodiodes based on the number of selected circuits to be programmed. Consequently, the interposer can be attached to a multi- or single-layer PCB interconnect based on their density.

- **Hardware interfaces** should support both optical and electronic communications. To manage the connection of the fiber arrays on the chip and optical sources/monitors we need to use an optical switch (Polatis switch in our case). This switch device should be properly configured, and its optical ports should be correctly paired with the optical sources/monitors and photonic chip ports using optical fibers. This process requires bookkeeping and proper usage of fiber connectors.

  For the electronics, the packaged chip is connected to the EIC boards via its PCB interconnect using 40-pin flex cables. Each EIC board is equipped with its own dedicated single-board computer (BeagleBone) functioning as a controller, and these are interconnected through SPI connections. For

high-speed communications, the PCB interconnect is linked to RF sources and monitors using up to 2×12 coaxial cables. As a result, the input/output interfaces include up to 24 RF connections (for modulators and high-speed detectors) integrated into the high-density interposer. Lastly, all controllable system components are accessible through a local Ethernet network.

- **Software interfaces:** Multiple software layers have been implemented to interact with our modular hardware setup. The system can be accessed through command-line interfaces (CLI), remote procedure call (RPC)protocols, or Application Programming Interfaces (APIs), making it adaptable and user-friendly for various applications and user expertise levels.

In this system architecture design, we have tried to standardize the interfaces and implement modular control electronics. We had the ambition to standardize the input-output interfaces as much as possible. This includes both optical and electrical connections, and enhancing compatibility across different modules and hardware. By adopting common design standards, such as the grid layout for the chip design and high-density interposers for electrical interfaces, components can be easily swapped and updated without extensive redesigns.

Moreover, our control architecture is meant to be modular, allowing us to connect multiple Electronic Interface and Control (EIC) boards through a local Ethernet network to a master controller. This setup enables local control tasks to be handled directly on the EIC boards while global control tasks are managed centrally. Since we only used one EIC board for the measurements in this work, the master controller was not implemented.

In the following sections, we describe how the demonstrator circuits have been organized on the chips. Next, we will briefly explain the fabrication process and packaging of the demonstrators and describe the electronic hardware developed to control the circuits. At the end, we elaborate the photonic-electronic-software stack defined to operate such an electro-optic system.

4.2 Chip Management

To showcase and compare programmable circuits, both re-circulating meshes (FPPICs) and application-specific PICs (ASPICs) such as switches, beamformers, and microwave circuits were designed on MORPHIC RUN2 and RUN3 chips. Implementing these two types of circuits not only showcases the capabilities and performance of the MEMS platform but also highlights the versatility of re-circulating meshes for general-purpose programming and their precision in replicating the

functionalities of ASPICs.

It is important to note that while each circuit must be designed on the chip, this is only part of the process: the interfaces connecting each circuit to the external environment also need to be defined and implemented in the software to enable proper control of the circuit, which is not a trivial task.

4.2.1 Application-Specific PICs (ASPICs)

For ASPICs, three distinct switch matrix architectures were explored and implemented, as detailed in Chapter 3: Benes, PI-loss (Path Independent Loss), and Crossbar, each with its own unique characteristics and applications.

We also implemented two types of RF-filters: IIR (Infinite Impulse Response) filters based on coupled resonators and hybrid filters based on Mach-Zehnder interferometers (MZI) loaded with resonators. These filters were designed by VLC Photonics, one of the partners in MORPHIC. Additionally, we explored optical beamforming circuits for applications in LiDAR and free-space optical communications. Two circuits were designed by Dr. Umar Khan: The Tree-based and Tap-based beamsteering networks. These networks are designed to control the power distribution and phase of the output beam, allowing for precise steering of the optical beam. The implementation of monitoring circuits within these beamforming networks enables accurate control and feedback for the beam steering process.

4.2.2 Field Programmable PICs (FP-PICs)

In Chapter 3, as detailed, we designed FP-PICs comprising a 24-cell MEMS-based mesh (RUN2), a 7-cell heaters-based mesh (RUN2), and a 126-cell MEMS-based mesh (RUN3). The initial aim for RUN2 was to create two identical meshes, one utilizing heaters and the other MEMS. However, near the fabrication deadline, we received more compact MEMS couplers, prompting us to incorporate additional blocks into each PIC cell. Hence, we could fit a 24-cell mesh instead of the 7-cell mesh within the allocated space on the chip.

4.2.3 MORPHIC Chips Overall Layout

Figure 4.2 shows the global chip layout for the MORPHIC RUN2 and RUN3. And here is the summary of the implemented circuits:

- FP-PIC Circuits (RUN2): A heater-based 7-cell radial-shape mesh and a

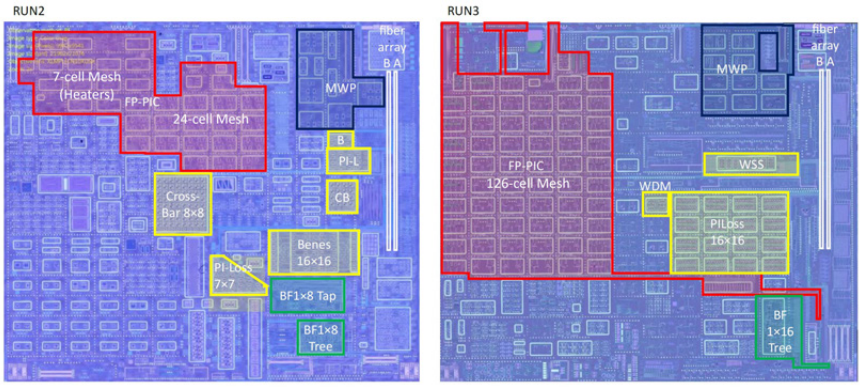


Figure 4.2: Demonstrator circuits on MORPHIC RUN2 and RUN3. The different circuits relate to switching (yellow), beamforming (green), microwave photonics (navy) and the FP-PIC (red). Each of these circuits is connected either to Fiber Array A or B.

MEMS-based 24-cell mesh with a custom shape.

- FP-PIC Circuits (RUN3): A 126-cell MEMS-based parallelogram-shaped mesh connected to a 1×16 phased array antenna via a beamformer control unit, and small test circuits for the full (3 couplers, 3 phase shifters) and custom nodes (3 couplers).

- Switches (RUN2): Three 4×4 mini switches (Benes, PI-loss, Crossbar), A Benes 16×16, A Crossbar 8×8, A PI-loss 7×7.

- Switches (RUN3): A PI-loss 16×16, and a wavelength selective switch (WSS).

- Beamformer (RUN2): 1×8 (tab- and tree-based).

- Beamformer (RUN3): 1×16 (tree-based).

- Microwave circuits, consisting of a modulator, a wavelength filters and photodetector (RUN2), and a multi-channel version of the same structure (RUN3).

Because of the failure of the MEMS circuits (see more details in Chapter 6) and a delay in receiving the packaged chips, our measurements were limited to the FP-PICs, Three 4×4 mini switches, and the Benes 16×16 switch.

4.2.4 Bondpads Grid System

The bondpad grid used in the project is a crucial element of the chip management, packaging and assembly process for the photonic integrated circuits. It plays a key role in interfacing the silicon photonics chip with external circuitry, which is essential for the functionality of MEMS components, sub-circuits, and circuits on the chip.

As shown in Fig. 4.3, the bondpad grid is organized into a regular grid pattern, which is divided into unit cells measuring $1500 \times 1050\ \mu\mathrm{m}^2$ each. Each cell is designed to accommodate up to 16 bondpads located on the West and South edges of the cell, facilitating connections to external circuits. The bondpads are spaced at $150\,\mu\mathrm{m}$, allowing for precision interfacing with external components through flip-chipping.

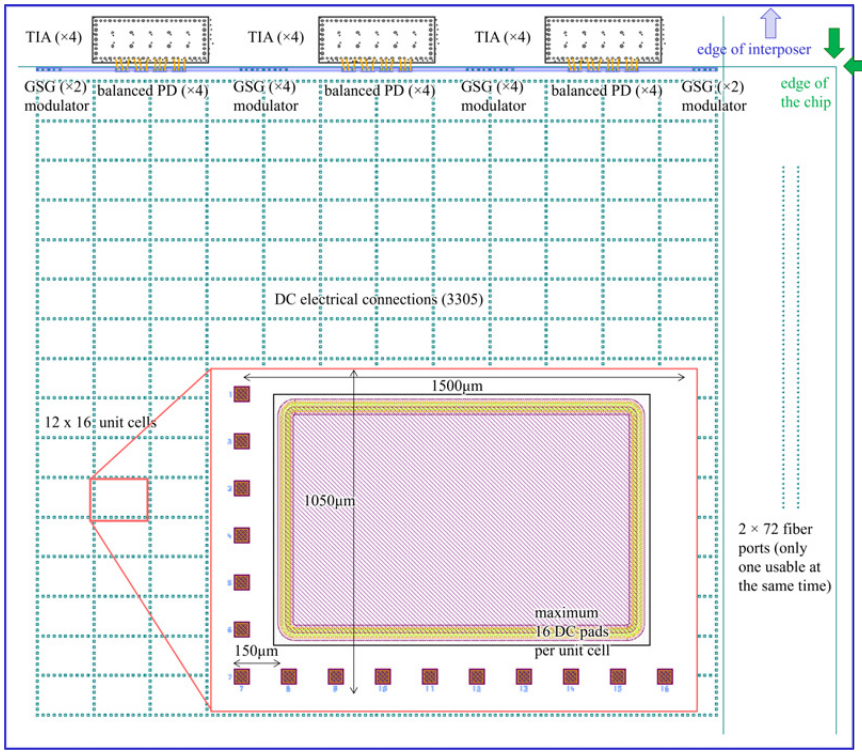


Figure 4.3: Overall layout of the interfaces of the MORPHIC RUN2 (and RUN3) chip. The chip consists of unit cells of DC connections that can interface with the interposers.

4.2.5 Fiber Arrays

Designed circuits on the chip are all connected to the fiber arrays near the edge of the chip. Figure 4.4 shows the map of the shunt waveguides across the fiber arrays, including the numbering of the fibers, as well as the corresponding position of the circuits on the fiber array, where input and output ports are indicated by green and red numbers, respectively. Since the Polatis switch can only accommodate 32 fibers at a time, each measurement session is dedicated to either the three 4×4 mini switches or the larger circuits, such as the 16×16 switch or the 126-cell FP-PIC.

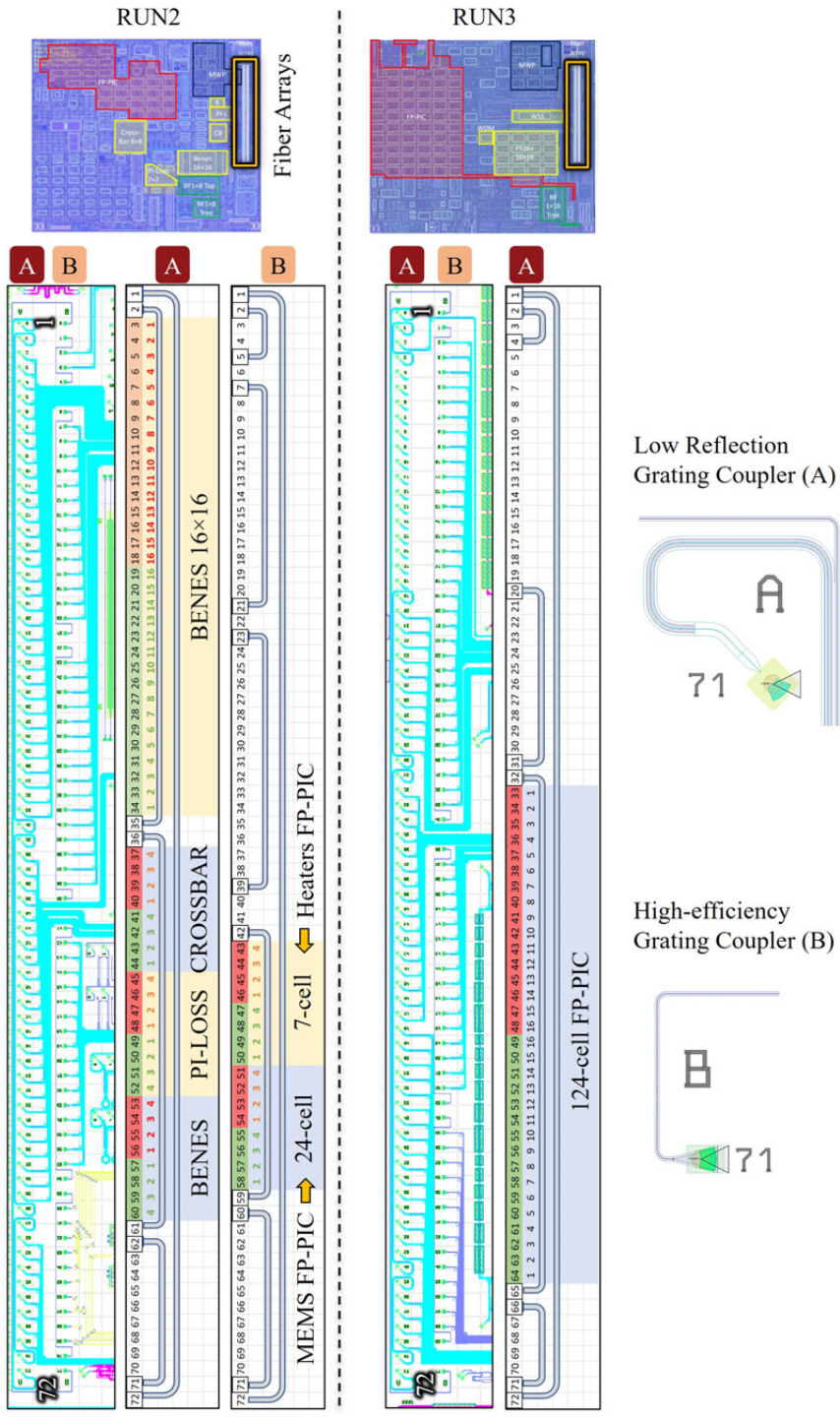


Figure 4.4: Location of the fiber array A on RUN2 and RUN3 chips and the maps of the shunt waveguides and circuit ports on the fiber arrays for each RUN.

4.3 Fabrication and Packaging

To ensure that our photonic chip is correctly interfaced with our electro-optic system components, appropriate packaging is essential. In this section, we will briefly overview the packaging process and key fabrication considerations. It's worth noting that the packaging process has been carried out by Tyndall after KTH has done the wafer-level sealing.

As starting point, we start with a simplified overview of the key steps involved in the fabrication and packaging flow (Fig. 4.5):

Wafer-Scale Silicon Photonics Fabrication (iSiPP50G Process by IMEC)**:** This initial step involves the standard fabrication flow for IMEC's $200\,\mathrm{mm}$ silicon photonics process, with customizations to support silicon photonics MEMS (Micro-Electro-Mechanical Systems). This stage defines all photonic integrated circuit components on the wafer.

Wafer Testing: After the initial wafer-scale processing, the wafers are tested to characterize passive performance before proceeding to post-processing. This helps in identifying any issues early on.

Coring and Wafer Cutting: The $200\,\mathrm{mm}$ wafers are reduced in size by cutting out two $100\,\mathrm{mm}$ wafers. This process is outsourced and is followed by edge grinding. The coring is crucial for handling and further processing.

MEMS Post-Processing: This step involves additional processing performed by EPFL to create movable MEMS structures on the chip [108]. It includes depositing a protective alumina layer, patterning, and releasing the MEMS by removing silicon oxide surrounding the structures using a vapor-phase hydrofluoric (HF) acid etch.

Sealing: Wafer-level sealing is performed at KTH by attaching a $100\,\mathrm{mm}$ SOI (Silicon On Insulator) lid wafer to the processed wafer using thermocompression bonding [109]. This step encapsulates the MEMS structures, protecting them from external environmental factors.

Wafer Testing (Post-Release): After MEMS release and sealing, the wafers undergo another round of testing to evaluate the performance of the released and sealed MEMS devices.

Dicing: The $100\,\mathrm{mm}$ wafers are then diced into individual chips, preparing them for packaging.

Stud Bumping: To facilitate solder bumping and overcome height differences caused by the lids, stud bumps are added to the bond pads.

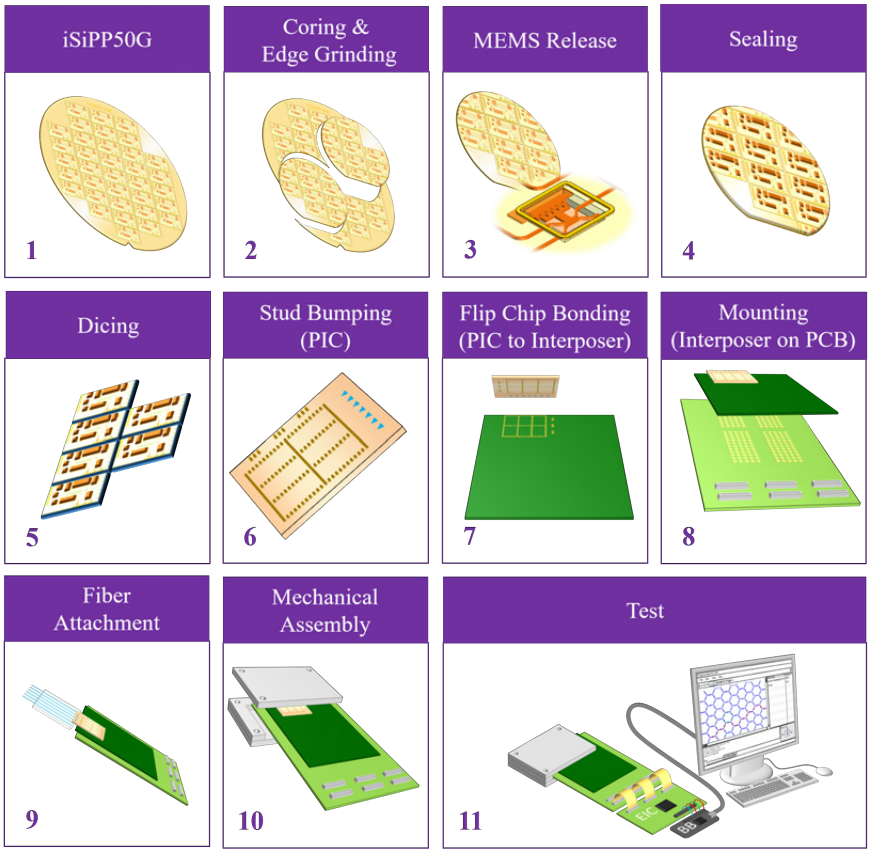


Figure 4.5: Fabrication and packaging main steps.

Interposer Fabrication: Depending on the type of demonstrator, an interposer is fabricated to fan out the dense electrical interconnects. We have implemented different types of interposers used, such as single-layer glass/silicon interposers and multi-layer ceramic interposers.

Flip-Chip Bonding on Interposer: The silicon photonics chip is flip-chipped onto the interposer using jetted solder balls and permanently fixed with an epoxy underfill.

PCB Mounting: The interposer is mounted on a printed circuit board (PCB), with the method depending on the interposer technology.

Fiber Attachment: Fiber arrays are actively aligned and glued to the chip, facilitating optical connections.

Mechanical Assembly: The assembly is packaged in a metal housing, which also acts as a thermal heat sink.

Final Testing: The assembled chips are tested to evaluate their performance.

It is important to emphasize that this process involves numerous design tasks, often handled by different teams or individuals, making effective communication and collaboration essential. The various components that require dedicated design efforts include:

- Photonic chip design
- Driver electronics design
- Interposer design
- Lid wafer design
- Printed circuit board (PCB) design
- Software development for system control

Given the complexity and interdependence of these elements, well-defined interfaces and seamless exchange of information between the different teams are crucial to ensure successful integration. A significant portion of my work focused on managing these interfaces and facilitating the flow of information across the design and test levels, ensuring that all components function harmoniously within the overall system. In addition to being involved in the design of chip circuits, single-layer interposers, and PCBs, I have also been responsible for cross-checking multilayer PCBs and interposers, providing feedback on EIC boards and MEMS designs based on photonic circuit specifications, and testing electronic hardware.

As will be discussed later, all designs are compiled into netlists, a critical part of our system and one of my key responsibilities in the MORPHIC project. A single error in defining the connections between an actuator ID on the mesh schematic and its corresponding physical bondpad on the chip can have a significant impact on the configuration process. For example, such a mistake could inadvertently convert a single-ring loaded MZI into a double-ring loaded MZI, create short-circuits, or leave certain building blocks unaddressable, causing unintended behavior.

4.3.1 MEMS processing

The MORPHIC project builds upon IMEC's established iSiPP50G silicon photonics platform by integrating suspended photonic MEMS components. Since

these devices are non-standard, additional post-processing steps are required after fabrication at IMEC. In this section, we briefly review the developed process flow that enables the release of the MEMS components while preserving the integrity of other platform elements, such as high-efficiency grating couplers and metallization for contact pads. We also present the fabricated components utilized to construct our circuits.

We should mention that this stage of the MORPHIC project has been carried out by our colleagues at EPFL. To our knowledge, the presented process is the first demonstration of enabling large scale silicon photonic MEMS production in a foundry platform.

The primary objective of the MEMS release process is to undercut the MEMS structures by removing the BOX layer beneath the silicon device layer (DL), allowing them to become freestanding and capable of movement. To avoid stiction [1] vapour-phase HF (VHF) is used. This etchant is highly aggressive and attacks most materials indiscriminately, requiring a robust protection and passivation strategy to safeguard most of the sample (e.g., the BEOL stack) while exposing only the MEMS cavity regions. We utilize alumina (Al_2O_3) as a passivation material, selectively removing it in areas where we need either to make direct contact with metal bond pads for characterization or to create access points for VHF to remove the BOX layer beneath the MEMS. Once the alumina is patterned, the final release etch is performed.

As illustrated in Fig. 4.6, the simplified process flow can be broken into the following steps:

- **Planarization and Filler Oxide removal:** The oxide between the waveguides, which is planarized during the iSiPP50G process, is first removed using a brief, timed buffer-HF (BHF) wet etch. The duration must be kept short, as the etch also affects any other exposed oxides.

- **Alumina Passivation:** The entire chip is passivated with alumina through atomic layer deposition (ALD), which conformally covers the top surfaces and sidewalls of the exposed waveguides.

- **Alumina Patterning:**

  - The alumina is opened over the AlCu bondpads only using a dry etch process.

[1] A failure mechanism where structures permanently adhere to a surface after drying of a liquid etchant.

  - In a second patterning step, the alumina is opened over the MEMS waveguides, using a dry etch and an additional wet BHF step, to avoid damage to the waveguide surfaces.

- **VHF Release Etch:** The exposed buried oxide is removed using a timed vapour HF etch

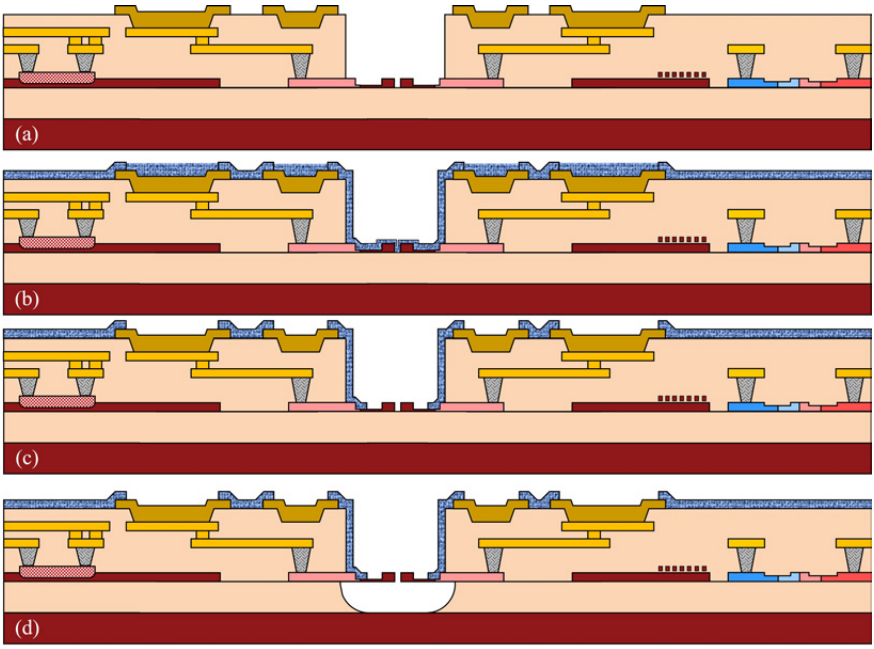


Figure 4.6: Process flow, developed by EPFL, for the MEMS post-processing detailing the primary steps of a) Planarization and filler oxide removal by BHF. b) Alumina passivation. c) Alumina patterning over the metallization and MEMS cavities by dry and wet etching. d) VHF release etching of the BOX.

Figure 4.7 shows three different PIC cells of the 126-cell FP-PIC fabricated on RUN 3. The pictures are taken from an unprocessed chip which corresponds to the step one of the mentioned process flow and the MEMS structures are not released. Position of each PIC cell is also illustrated by the circuit map in the figure. Also, as expected, the substrate grounder membrane is clearly seen since it has not been released yet.

As illustrated in Fig. 3.23 in section 3.2.3.1, we have used coupler R3A and phase shifter R3B to construct the circuit mesh. These components operate based on the in-plane actuation mechanism. It's worth noting that in-plane actuators tend to be larger and have lower out-of-plane stiffness compared to out-of-plane

actuators. However, comb-drive actuators offer the advantage of enabling larger, reliable displacements, as the pull-in effect is largely avoided. Based on tested structures from our partners in RUN2, the most reliable performance was achieved using in-plane actuators overall. Notably, key reliability features unique to in-plane actuators include hysteresis-free actuation, mechanical stoppers, and grounding of movable parts.

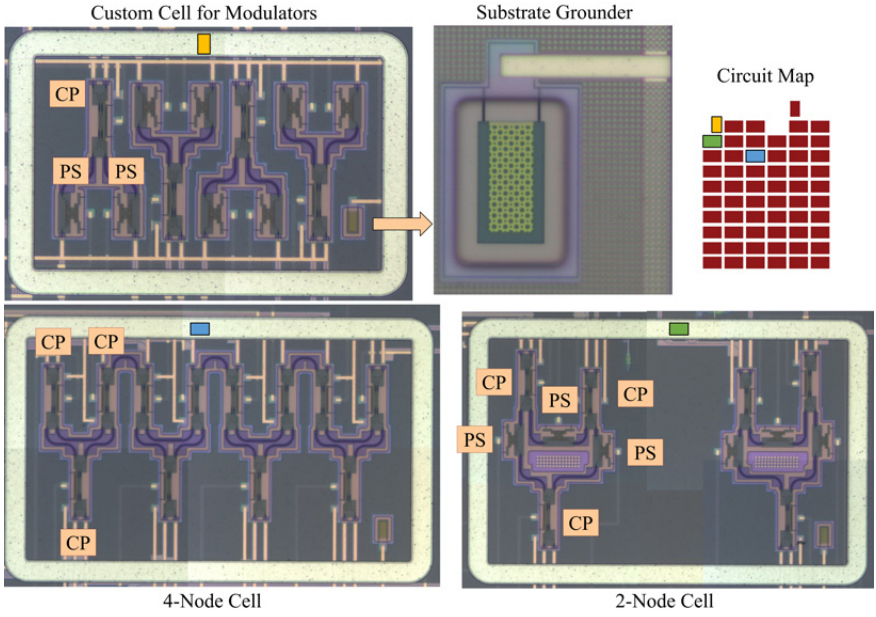


Figure 4.7: Three distinct PIC cells from the 126-cell FP-PIC on an unprocessed chip from MORPHIC RUN3. The circuit map shows the location of these cells on the circuit mesh. We also have shown the zoomed-in image of the substrate grounder membrane of the custom cell for modulators.

We also have shown a PIC cell from the 24-cell FP-PIC (RUN2) and a custom PIC cell from the 126-cell FP-PIC (RUN3) in Fig. 4.8 and Fig. 4.9, respectively. These images are taken from etched chips corresponding to the step 4 of the process flow.

The demonstrated PIC cell of the 24-cell FP-PIC is composed of two full nodes, each consisting of 3 couplers, 3 phase shifters, and a substrate grounder. The nodes were constructed using phase shifter R2A (Fig. 3.23c), which operates based on the out-of-plane actuation mechanism. To provide a clearer view of the released devices, we have included a close-up image along with zoomed-in views of two different suspended components of this actuator showing long suspended waveguides.

Long suspended waveguides, such as spirals, can help reducing the required applied voltage for devices, but they also increase the footprint and make the devices more susceptible to collapse. Hence, using s-bend waveguides improves the device performance, while still keeping the actuation voltage below the 40 V upper limit. In fact, characterization of RUN2 devices revealed that those with long suspended waveguides were prone to collapse, leading to modifications in the default building blocks for RUN3, such as using phase shifter R3A (Fig. 3.23e) instead of phase shifter R2A.

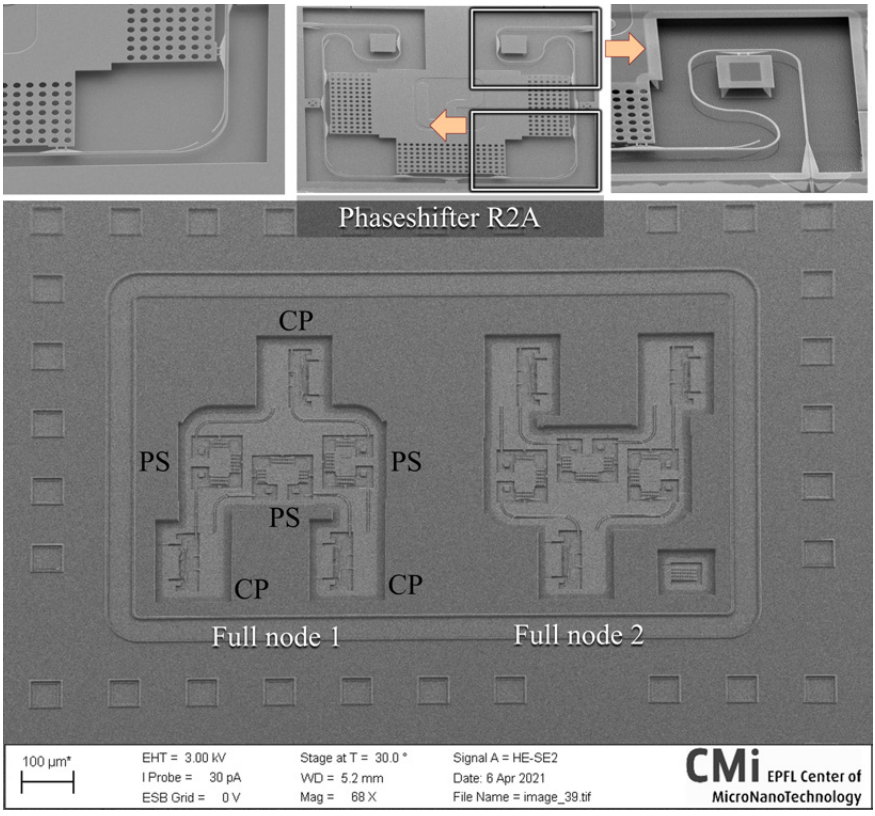


Figure 4.8: SEM image of a PIC cell of the 24-FP-PIC following the release of the MEMS, showcasing two full nodes. The zoomed-in images highlight one of the phase shifters used to construct the nodes, along with its two suspended waveguides at different locations. (Image Credit: EPFL, KTH)

For our final demonstration of the FP-PIC cells, we present a custom PIC cell from the 126-FP-PIC located in the northeast, adjacent to the circuit's SOAs. This cell features three custom nodes constructed with three couplers. The first sign of the etching process is evident through a color change in the substrate grounder

membrane, indicating its collapse after applying a 0-5V voltage step. Additionally, we provide two zoomed-in images of the north ports of the top left coupler in the 3rd node of the PIC cell. They highlight the suspended rib waveguide, used for inter-cavity connections of the MEMS as described in Chapter 3, along with its trenches and the suspended waveguides inside the MEMS cavity.

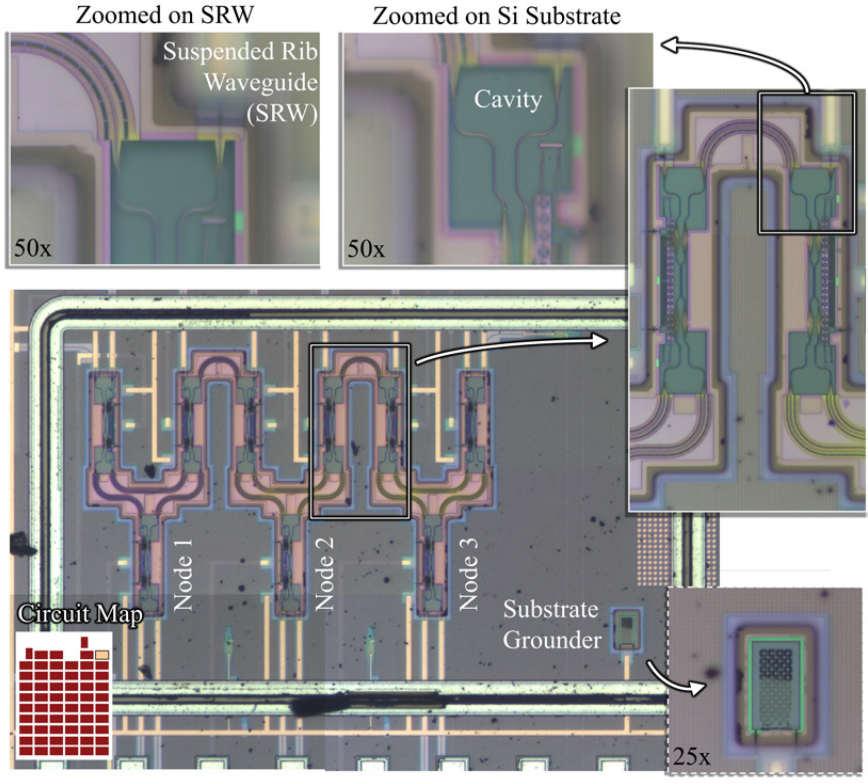


Figure 4.9: A customized PIC cell of the 126-cell FP-PIC including 3 nodes. Zoomed-in images on the top show the suspended rib waveguide, used for inter-cavity connections of the MEMS, along with its trenches and the suspended waveguides inside the MEMS cavity. The color change of the substrate grounder indicate the successful release of the membrane and also its deliberate collapse.

4.3.2 Hermetic Sealing

One of the important transient steps between the fabrication process and packaging of our photonic chips is the wafer-level hermetic sealing of the MEMS devices (Fig. 4.10). In this process, the sealing caps are integrated with the MEMS structures

during the post-processing of silicon photonic integrated circuits. The main purpose of this step is to protect the delicate MEMS structures from external factors that could degrade their performance over time. This is particularly important for ensuring the reliability and longevity of the integrated silicon photonic MEMS devices. In fact, the hermetic sealing process aims to achieve a high level of hermeticity, meaning that the sealed MEMS structures are effectively isolated from the external environment. This prevents the ingress of moisture, dust and particles, and other harmful substances that could lead to corrosion, oxidation, or mechanical degradation of the MEMS components.

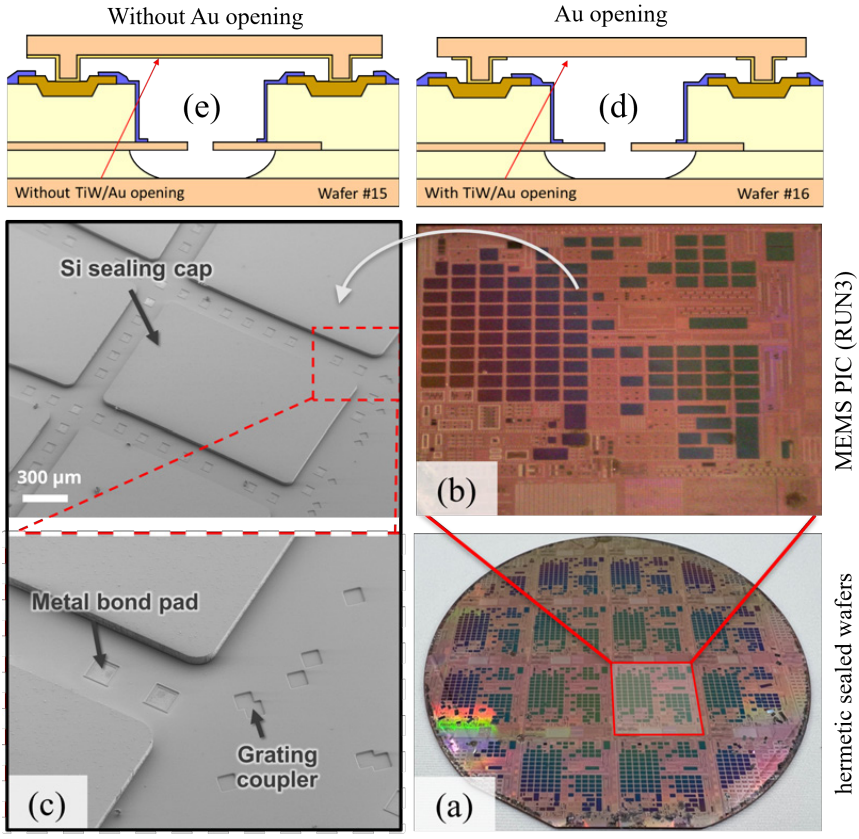


Figure 4.10: a) An example of wafer-level sealing for a b) MORPHIC RUN3 chip, where c) silicon caps have been used for the lids. The silicon sealing lids d) without and e) with Au opening. Au opening enables infrared imaging thought the lids. (Figure a and c are provided by KTH)

One notable aspect of the wafer-level hermetic sealing process is the use of thin

sealing caps, which are applied at the wafer level. These caps are designed to be very thin, with thicknesses on the order of tens of micrometers ($< 30\mu m$). The thinness of the sealing caps is essential for enabling subsequent electronic-photonic assembly steps without significant height disparities or challenges in alignment and integration. Additionally, a layer of TiW/Au deposited along these sealing rings serves as a contact material with the aluminum metal ring on the photonic device wafer during thermo-compression bonding.

Since covering the lids entirely with a gold layer on the inside (Fig. 4.10e) prevents infrared imaging through the lids, a revised version of the lid was introduced in RUN3 with openings in the gold layer (Fig. 4.10d). These openings are strategically designed to allow for infrared imaging through the silicon lids, enabling inspection of the MEMS devices without having to remove the lids. This design modification allows for the non-destructive inspection of the devices, utilizing the property of silicon being transparent in the infrared wavelength range, thus facilitating easier quality control and assessment of the MEMS structures post-sealing.

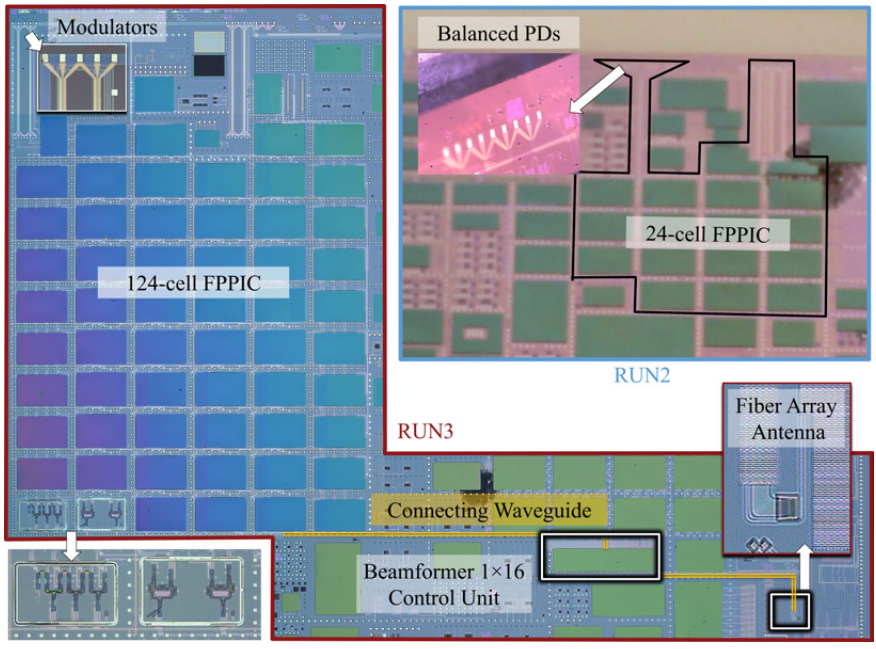


Figure 4.11: Sealed FP-PIC circuits from MORPHIC RUN2 and RUN3.

Figure 4.11 displays two sealed FP-PIC circuits from MORPHIC RUN2 and RUN3. Despite the difference in cap sizes, they are successfully implemented. The caps over two PIC cells of the 126-cell FP-PIC in the southeast corner have been removed, revealing their nodes. We also highlight the ASPIC circuit connected

to the 126-cell FP-PIC mesh, as well as the modulators located at the top of the circuits. In the 24-cell FP-PIC, a dark spot indicates where an VHF attack occurred during the post-processing flow.

4.3.3 Assembling Schemes

In MORPHIC, we applied two distinct approaches for assembling the photonic integrated circuit (PIC) demonstrators: mini demonstrators and full demonstrators. Each approach has different objectives and is based on the type of the interposer selected for the assembly. Here's a comparison of the two based on the fabrication flow and their intended applications:

Mini demonstrators: They utilize a *single-layer interposer* made of either glass or silicon and are suitable for less complex PICs. The interposer and PCB need customization for each individual demonstrator, which could lead to higher per-unit costs for prototypes but allows for specific design adjustments. The process is relatively simple, involving breaking out a selected number of bondpads from the chip and connecting them to a custom PCB with wirebonds at the edge. This simplicity can be advantageous for rapid prototyping and testing of concepts. Two mini demonstrators have been assembled: version1 using a glass interposer and version2 using a Si interposer.

Full demonstrators: They make use of a *multi-layer ceramic interposer*, which can handle the complexity of breaking out all 3305 possible bondpads from the photonics chip to a larger pitch. This complexity allows for more extensive connectivity options. While the ceramic interposer itself might be more complex and costly, it enables the use of a low-cost, standard PCB to connect only the relevant pads, potentially reducing overall costs for larger runs.In fact, this interposer is useful for any future design that places its bondpads on the predefined grid. The process is more complex due to the nature of the multi-layer ceramic interposer and the comprehensive connectivity it supports. This complexity is necessary for large-scale, complex circuits. Hence, our full demonstrators support all the demonstrator circuits implemented on the PICs. For this thesis, only one version of the full demonstrators (RUN2 chip) was available for the measurements.

For comparison, the schematics cross-section representation of both full and mini demonstrators are demonstrated in Fig. 4.12. As seen, one major difference is that in mini demonstrators the PIC is faced down, while in full demonstrators it is faced up. The following two subsections will delve into the design and assembly processes for both the mini and full demonstrators. As the packaging and assembly of the PICs fall beyond the scope of this thesis, only the main steps are briefly outlined.

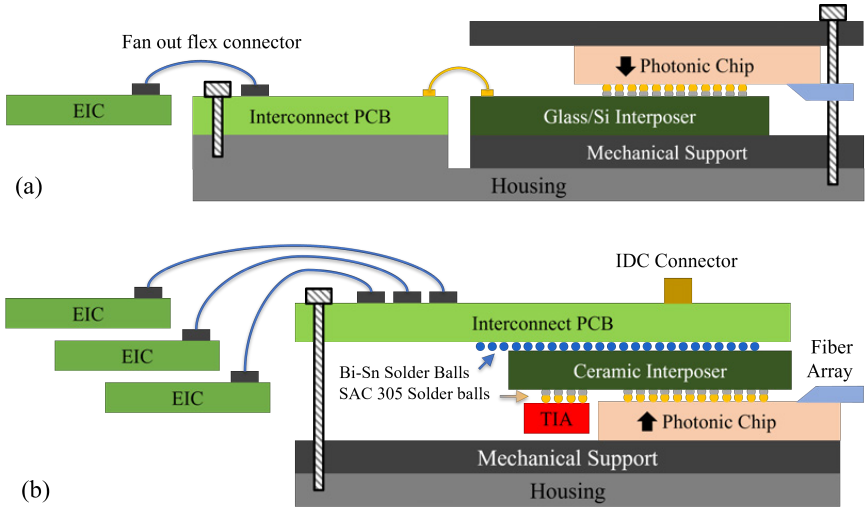


Figure 4.12: Schematic cross section representation of a) mini demonstrators and b) full demonstrators.

4.3.3.1 Mini Demonstrators

The idea behind the mini demonstrators is to break down the complex full demonstrator into smaller groups, each centered around specific optical circuits on the silicon chip. These groups were also organized in a way that allowed electrical connections to be made through a single-layer interposer, making this approach suitable only for moderately sized circuits.

As shown in Fig. 4.13 I designed six single-layer interposers for circuits requiring significantly fewer connections compared to 3305 available on the full-scale interposer. This makes the design and production process distinctly different from that used for full-scale interposers. Our interposer designs include 7-cell FPPIC (heaters), 24-cell FPPIC, microwaves, beamformers, switch A ($crossbar4 \times 4$, $PI - loss4 \times 4$, $benes4 \times 4$, and $benes16 \times 16$), and switch B ($crossbar4 \times 4$ and $benes16 \times 16$) circuits. The purpose of the single-layer interposers is to route the connections from the PIC's flip-chip pads to wire-bond pads at its edge. And, they have three pad regions: flip-chip pads (ex: red and yellows boxes for FPPIC heaters and MEMS in Fig. 4.13), pads for mechanical stability (unused pads on the pads grid), and wire-bond pads which are located close to the border of the interposer and will be connected to the PCB pads using wire bonding.

In addition of single-layer interposers, I have designed their dedicated interconnect PCB using Kicad interfaced with python codes (Fig. 4.14). It is worth to

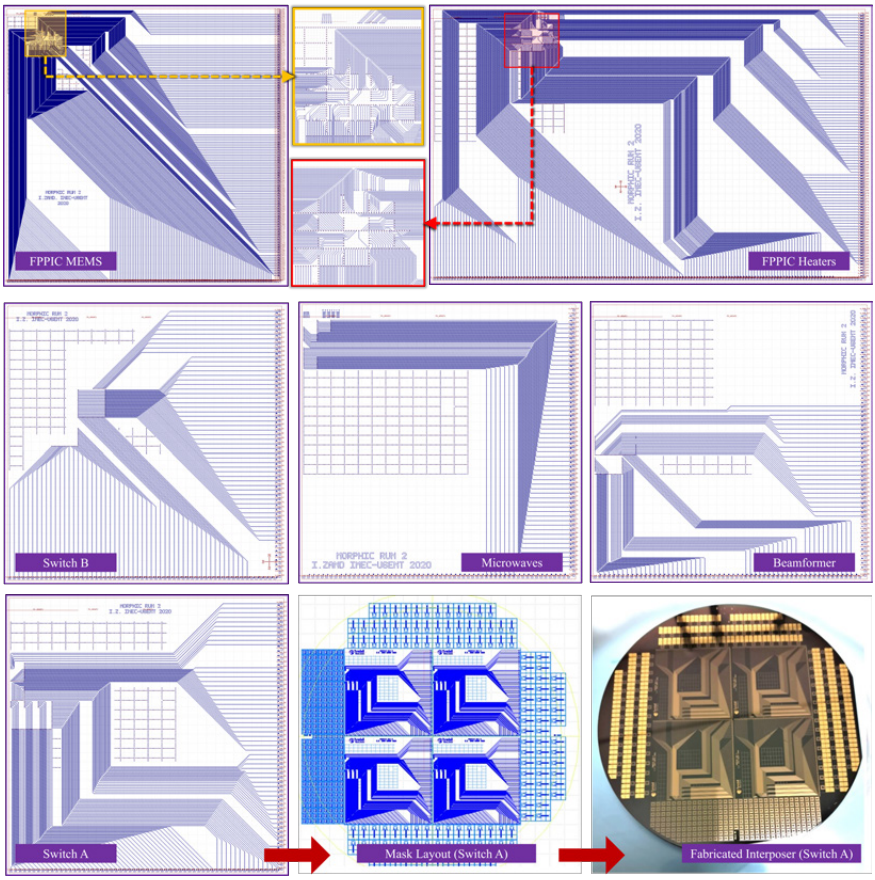


Figure 4.13: The small-scale interposers designed for mini demonstrators. The interposers are made of a single Au layer over a 100 mm *Si wafers at Tyndall silicon fab lab. The last row shows as an example of the completed layout for the switch A demonstrators and its fabricated Si interposer.*

mention that pitch size of the bond-pads on the edge of interposer dictated by the interconnect PCB pads which are $400\mu m$.

In the following sections, we elaborate a series of steps involved in packaging photonic integrated circuits (PICs) for mini demonstrators. This part outlines a specialized process tailored to package smaller scale or less complex circuits using simplified approaches that still maintain high precision and functionality. The implemented approach is similar for mini demonstrators using both glass and silicon interposers. The summary of the steps is illustrated in Fig. 4.15.

Here is packaging flow of the mini demonstrators after fabrication of the inter-

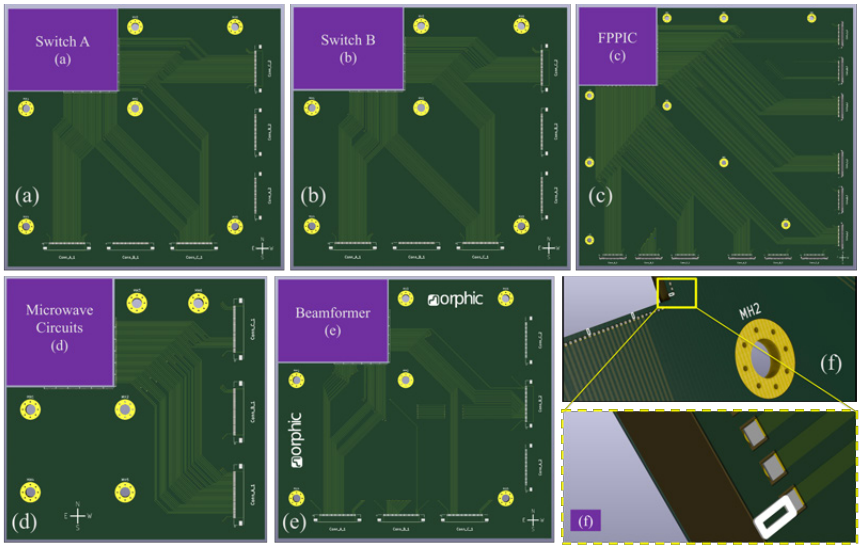


Figure 4.14: a-d) Designed interconnect PCBs for the single-layer interposers of mini demonstrators. f) enlarged view of PCB pads on its edge for wire bonding with the interposer bond-pads.

posers, a task primarily carried out by Tyndall:

Gold Stud Bumping and Coining (Silicon Photonics Chip): To create reliable, high-density electrical contacts on the photonic chip for subsequent attachment to the interposer. Gold stud bumping involves depositing small gold studs on the contact pads of the photonic chip, followed by coining, which slightly flattens the bumps to ensure better contact with the interposer. This step enhances the mechanical stability and electrical connectivity.

Solder Ball Jetting and Re-flow (Silicon Interposer): To prepare the silicon interposer for attachment by adding solder balls that will connect with the gold bumps on the photonic chip. Solder ball jetting is a precision deposition technique where solder balls are accurately placed on the interposer's contact pads. Re-flow is then performed to melt and solidify the solder, ensuring robust electrical connections.

Flip Chip Bonding (Silicon Photonics Chip to Silicon Interposer): To mechanically and electrically connect the photonic chip to the silicon interposer. The photonic chip is aligned and then flipped onto the interposer so that the gold bumps align with the solder balls. The assembly is then heated to reflow the solder, creating a permanent bond between the chip and the interposer.

Underfill Epoxy Filling: To provide additional mechanical strength and protect the bonded interface from environmental damage. After flip-chip bonding, an epoxy underfill is applied to fill any gaps between the chip and the interposer. This step is followed by X-ray imaging to inspect the integrity of the connections and ensure there are no defects such as voids or misalignments in the solder.

Mechanical Assembly (PIC, Interposer, and PCB): The mechanical assembly process for the full demonstrators RUN2-PCB1 and RUN3-PCB2 involved similar designs due to their matching dimensions and lack of RF connections. The housings were designed using SolidWorks software and manufactured from aluminum alloy. The assembly involved placing a thermally conductive graphite sheet beneath the silicon photonics chip for heat dissipation, mounting the PCB-interposer assembly onto a base plate, and attaching a 72-channel fiber array to the photonics chip using UV curable epoxy. A metallic lid was then secured over the grating coupler area to protect the fiber array, and a black rubber sleeve was shrunk over the fiber to prevent strain.

Fiber Array Attachment: Each MORPHIC chip has two sets of 72 grating couplers which are compatible with standard fiber arrays with a $127\,\mu\mathrm{m}$ pitch. The fiber attachment process involved using a high-precision auto aligner with motorized translation stages to align the fibers with the photonics chip, adjusting for yaw, pitch, and roll to ensure proper wavelength alignment. After alignment, the fiber array was fixed in place using UV curable epoxy and secured with a metallic lid, followed by a black rubber sleeve to protect the fiber from excessive bending.

Final Assembly: To complete the packaging process and prepare the mini demonstrator for testing or deployment. Involves sealing the assembly in a protective casing, performing final checks, and verifying that all components are securely integrated and functional. This step included measurements of coupling loss from channel 1 to channel 72 of the fiber array, with losses recorded at 11.42 dB for the Switch A mini demonstrator after curing the UV epoxies. An additional loss of 1 dB due to the UV epoxy was anticipated. Transmission losses from the packaged device were evaluated after 24 hours and showed no changes, confirming the stability of the Switch A package.

We should mention that first glass versions were made by EPFL, but the gold layer was not thick enough, which resulted in dissolved bondpads and open circuits. The second version was made by Tyndall on a silicon substrate with thicker gold ($2\,\mu\mathrm{m}$).

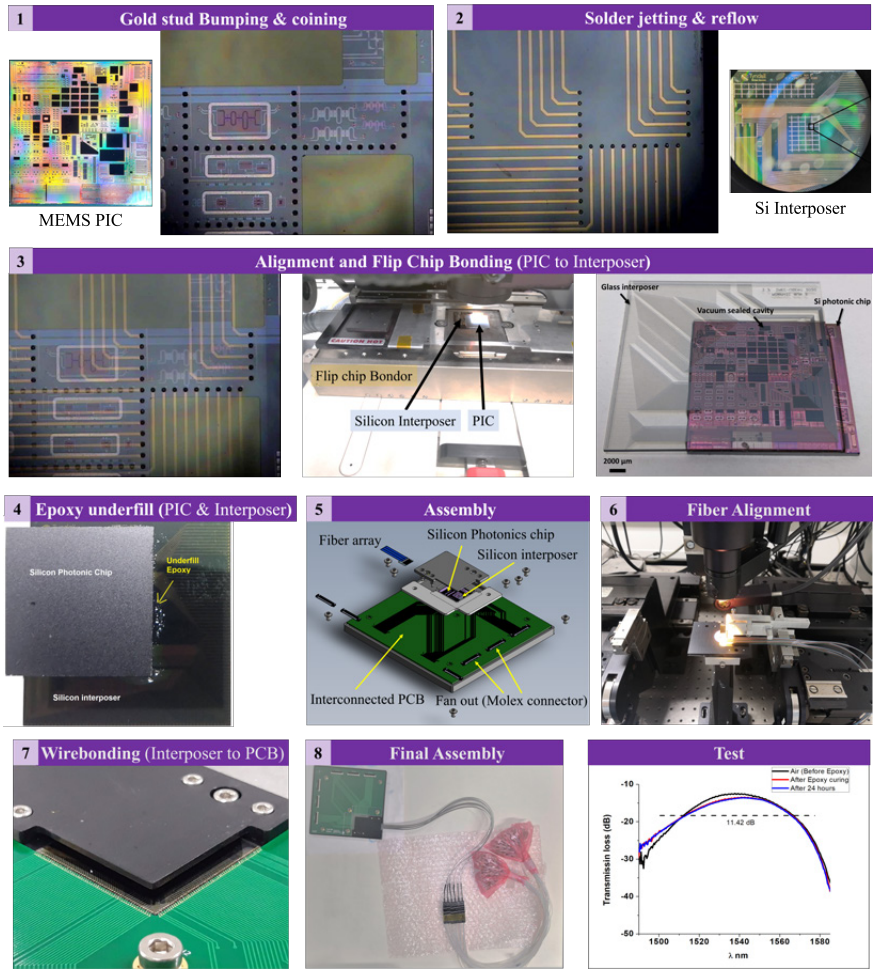
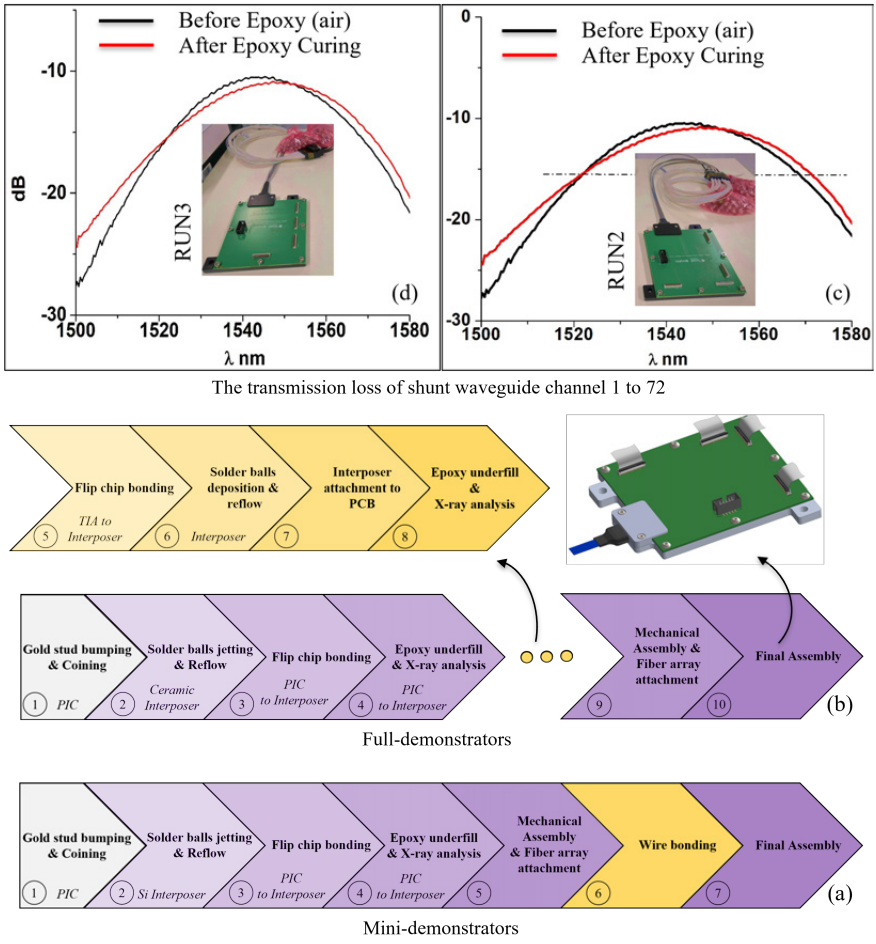


Figure 4.15: Packaging steps of mini demonstrators done by Tyndall.

4.3.3.2 Full Demonstrators

The packaging design for the full demonstrator uses a multi-layer interposer to accommodate 12 RF inputs, 12 RF outputs, 3305 DC electrical connections, and two sets of 72 optical connections (not at the same time). This 3D integration strategy, which combines optical and electrical connections, schematically illustrated in Fig. 4.12. As seen in the figure, the silicon photonics chip and high-speed trans-impedance amplifiers (TIAs) are initially flip-chipped onto the multi-layer ceramic interposer. The DC interconnections between the bondpads of the MEMS and TIAs are made using lead-free solder balls. Note that not all 3305 possible connections are made, but only those for which there are matching bondpads on the chip. It is important to highlight a design consideration; for mechanical stability, bondpads need to be distributed across the entire chip, not just in the areas of the circuits of interest. To achieve uniform coverage, dummy bondpads were added during the design phase to ensure proper balance across the chip. Subsequently, optical connections are established via 72-channel fiber arrays that interface with the on-chip grating couplers of the MEMS demonstrator circuits. On the other side of the interposer, specific DC pins are routed to a custom printed circuit board (PCB). This PCB hosts all 3305 DC pads, which are mounted using low-melting Bi-Sn solder balls, though only selected pads are extended to connectors for electronic interfacing. This extensive and intricate packaging process is carried out at Tyndall.

The packaging process for the full demonstrators follows a similar approach to that of the mini demonstrators, with differences stemming from the incorporation of the full-scale interposer and the interconnect PCB. Figure 4.16a-b illustrates and compares the key steps in the packaging processes for both the mini and full demonstrators, highlighting their distinctions. The completed packages of two full demonstrators are displayed in insets of Figs.4.16c-d. The coupling efficiencies for the fiber array attachments to these demonstrators are shown in Figs.4.16c-d. An SLED source was connected to channel 1 and channel 72 was linked to a photodetector. The transmission loss for the shunt waveguide on the silicon photonics chip was measured for both demonstrators. After curing the UV epoxies, the coupling losses were recorded at $11.32\,\mathrm{dB}$ for RUN2 PCB and $11.4\,\mathrm{dB}$ for RUN3 PCB per fiber pair, as indicated in Figs.4.16c-d. An additional loss of $1\,\mathrm{dB}$ from the UV epoxy was observed and anticipated.



The transmission loss of shunt waveguide channel 1 to 72

Figure 4.16: a-b) Comparison of packaging steps of mini and full demonstrators. c-d) Transmission loss measurement of the shunt waveguides 0 to 71 for RUN2 and RUN3 PCB samples. Complete package of full demonstrators (RUN2 and RUN3) are also shown as insets. These steps are done by Tyndall.

4.4 Electronic Boards and Hardware

Large-scale programmable photonic integrated circuits require many optical I/Os (72 in our case) and a substantial number of electrical I/Os (up to 1000 for the 126-cells FPPIC designed for RUN3) to control the MEMS and monitor photodiodes. The architecture of the control electronics utilizes commercial off-the-shelf components, which proves the capability to manage a large array of photonic

MEMS devices with existing technology. To implement hardware and software interfaces for accessing and programming large-scale PICs several key stages have been considered:

- The fine-pitch electrical connections on the photonic chip are expanded via a silicon/ceramic interposer to a size compatible with conventional printed circuit boards (PCBs).
- An interconnect PCB extends these electrical connections from the interposer to multiple EIC boards using high-density flex connectors.
- Each EIC board is equipped with 64 high-voltage drivers and 32 photocurrent readout circuits, which are digitally managed via DACs and ADCs.
- A standard single board computer (SBC), such as the BeagleBone, runs the software that interfaces with the EIC boards, enabling software control loops that adjust the MEMS driving signals based on the currents from the photodiodes.

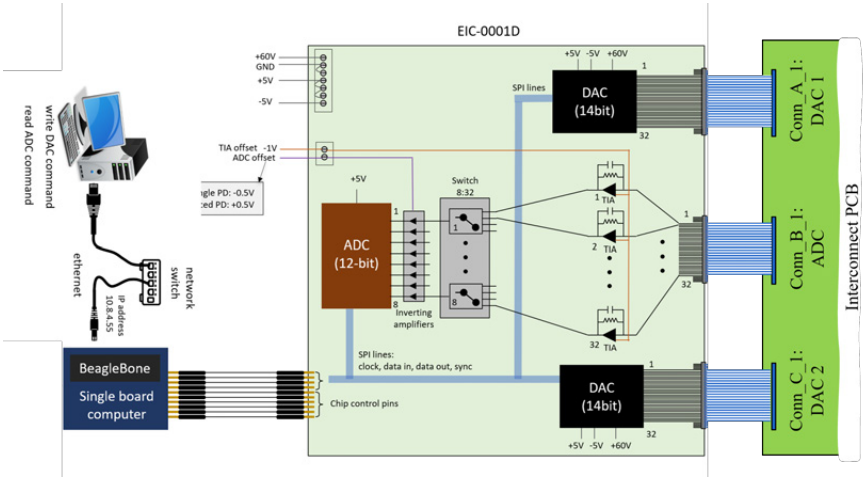


Figure 4.17: Control electronics for programmable photonic integrated chip made by Tyndall.

Figure 4.17 shows the high-level architecture of Electronic and Interface Control (EIC) Board which is connected to PIC through the interconnect PCB and interposer. The EIC board features two multichannel DACs (Digital to Analog Converter), which deliver a total of 64 programmable voltages ranging from 0 to 50V with 14-bit accuracy. These voltages are used to drive the MEMS via flex connectors A and C. Additionally, there are 32 TIAs for photodiode (PD) readouts connected

through flex connector B. Analog switches, which connect 4 inputs to 1 output, are employed to select one TIA from four, channeling it to one of the 8 ADC (Analog to Digital Converter) inputs. A specific SBC (BeagleBone board) facilitates the digital interfacing with the DACs and ADCs. Further details on the software communication architecture implemented on the BeagleBone and EIC boards are provided in Section 5.3.

The readout circuit path includes 8 analog switches, which are digitally controlled via 3 GPIO connections from the BeagleBone board. Utilizing a single 8-input ADC chip simplifies the management of 32 readout lines and streamlines communication over the SPI interface, which is also used by the two DAC chips. In the designs we used for the characterization, an inverter stage was incorporated just before the ADC inputs to enhance flexibility in photodiode (PD) configurations supported by the EIC boards. Additionally, external pins are available for setting the inverter stage offset voltage (also known as ADC offset) and TIA offset voltage. These features enable the use of the same TIA for both single-ended and balanced photodiodes.

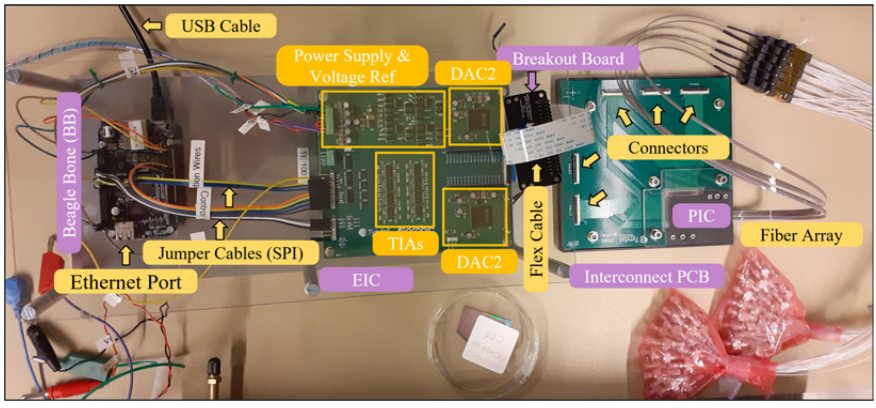


Figure 4.18: Control electronics for programmable photonic integrated chip.

Figure 4.18 shows the electronic control boards (EIC and BB) next to the packaged chip. As seen, the BB is connected to the EIC using jumper cables to provide SPI connection. And, the flex connectors connect EIC board to the interconnect PCB. We also use a breakout board to check applied voltages from the EIC board. The BB is powered by the PC via the provided USB cable, and command signals will be sent through the ethernet cable. And, the PC communicates with the BeagleBone, which is identified by a unique IP address, through a command-line interface (CLI) over an Ethernet connection. CLI commands can be executed using either Python or Matlab. We, also, need four power supplies for TIAs, DACs and ADCs digital power supply, and DAC Output amplifier high voltage supply ($60V$).

4.5 Design Trade-offs

In this section we highlight various trade-offs in chip design related to the integration of MEMS components and their interaction with other circuit elements.

- **Footprint Allocation:**
  - Large bond-pads, fiber coupler arrays, and alignment markers take up significant space on the chip. The spacing requirements for bond-pads ($150\,\mu\mathrm{m}$) and fiber couplers ($127\,\mu\mathrm{m}$) consume space that could otherwise be used for additional circuits.
  - The wide rims around the air-clad MEMS cavities do not contribute to functionality but are necessary for optical and electrical interfacing, consuming space that could be used for additional devices. It was $37\,\mu\mathrm{m}$ for circuits fabricated in RUN2, but the width reduced to $15\,\mu\mathrm{m}$ for RUN3 circuits.
  - The hermetically bonded silicon lid also requires additional space, which places constraints on photonic circuit placement. Related designs should be refined to make the process more tolerant of misalignment in later runs.
- **Circuit Construction:** Circuits can either be constructed from individual MEMS blocks in separate cavities or combined into subcircuits within a single cavity. The latter approach is limited by the inability to route metal wires within the cavity, though improvements were made in RUN3 by designing MEMS blocks with electrical interfaces on one side, enabling more complex circuits within a single cavity.
- **Pattern Density:** MEMS cavities cannot have dummies in metal and waveguide layers, which limits the pattern density and the size of MEMS cavities. Minimizing MEMS regions and incorporating density control features complicate the design but are necessary for fabrication and processing performance.
- **Grounding:** Since we are using silicon waveguides that connect an entire circuit together optically, all waveguides are at the same potential. Hence, we have to either electrically isolate all MEMS building blocks or keep all waveguides on ground potential.
  - **Isolated Building Blocks:** Electrical isolation of building blocks could reduce crosstalk but would require cutting a slot through the silicon waveguides, resulting in a non-negligible loss, even with precise engineering.

  - **Grounded Waveguides:** Maintaining all waveguides at ground potential prevents electrical crosstalk between the MEMS components, but it restricts the geometric design flexibility of the MEMS building blocks. We chose this option, and later measurement results confirmed it to be a good approach.

- **Actuators Control and Calibration:** In this work, we developed a modeling scheme to evaluate errors in programmable photonic circuits stemming from imperfections in phase shifters and tunable couplers. These imperfections can arise from various factors, but a key issue we focused on was the discretization error introduced by digital-to-analog converters (DACs). While high-resolution DACs result in fewer errors in phase shifts and couplings, they are also more expensive compared to their low-resolution counterparts. Therefore, the choice of DACs and control schemes should be tailored to the required accuracy of the mesh. Also, the calibration process improves precision but adds complexity to the control system, requiring careful error management to meet application needs.

4.6 Interfaces and Tools For Software Control

The last piece of our system architecture consists of the software layers enabling the reconfiguration of photonic circuits without needing physical modifications to the hardware. This capability is critical for rapid prototyping and deployment of new photonic applications, allowing users to adapt their hardware to different functions based on changing needs or experimental outcomes. The integrated control software also reduces complexity for the user, making advanced photonic circuits more accessible for research and industrial applications.

Figure 4.19 illustrates the photonic-electronic-software stack required for driving the reconfigurable circuits. As seen, our control stack has been divided to three major layers of *Framework (Software)*, *Hardware Driver (Electronics)*, and *Physical Layer (Photonics)*. The framework includes additional layers such as *Developer Kit*, *Programming*, and *Control strategies*. The electronics layer mainly addresses the interfaces, feedback loops, and command signals (read/write). And, the photonic layer is, in fact, the physical layer of photonic chips including the tunable components and photodetectors necessary for light manipulation and detection. Note that all these layers require some form of representation in the software stack itself, and every element on every layer needs to have a unique identifier.

In this section, we overview the control stack and further details are provided in the next chapter (5), where we discuss the software framework I developed for the

MORPHIC project.

4.6.1 Framework

The software framework which is discussed in details in Chapter 5 includes tools for circuit simulation, visualization, and graph-based routing. Also, it supports various scripting and programming functionalities, enabling data analysis, device configuration, and measurement automation. As shown in Fig. 4.19 it has three major layers of *Developer Kit*, *Programming*, and *Control Strategies* which are discussed in the following.

4.6.1.1 Netlist

One of the key components of the framework is the master netlist, which is created after all hardware layers—including interposers, PCB interconnects, and the EIC board—are designed. This netlist links the components shown in the schematic layout to the actual electrical pads on the chip's driving building blocks and to the connector pins on the PCB interconnect (see section 5.8). Utilizing this mapping, we have established an interface between the framework and the BeagleBone, facilitating automated command transmission to the DACs and ADCs.

Creating circuit netlists was a crucial aspect of my work and played a key role in system design and management, as it outlines the connections across the entire system. It was an iterative and cumbersome process and required constant communication with Tyndall to make sure all the physical connections and routes are according to the design layouts. For example, only for 126-cell FP-PIC circuit connectivity of 1000 bondpads on the photonic chip had to be cross checked on different layers of the system.

Further information on the netlists, along with a visualization of the process, is provided in Chapter 5, Section 5.8, which covers the data management aspects of the software framework.

4.6.1.2 Developer Kit

For photonic chips, similar to traditional electronic chips, a comprehensive *developer kit* is crucial to enable efficient and effective design, development, and integration of photonic components and systems. In our architecture, we have considered three parts of *Programming Environment*, *Modular Components*, and *APIs* for this layer.

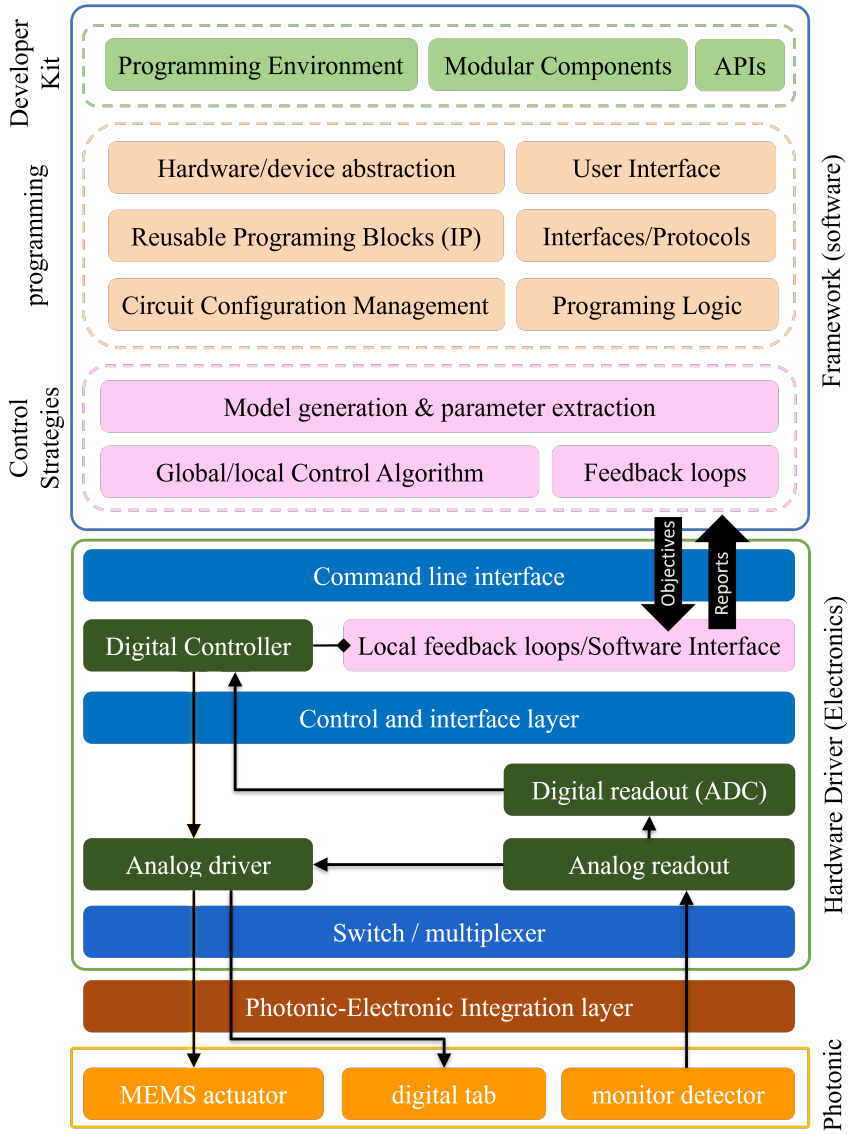


Figure 4.19: a) Photonic-electronic-software control stack for the photonic MEMS and FP-PIC platform.

The *programming environment* for photonic chips includes the software tools and a framework necessary to write, test, and deploy code that interacts with and controls photonic hardware, simulations, and design processes. For this end, we have used PyCharm and python as our IDE and programming language, respectively. Additionally, we have used IPKISS (by Luceda) for the layout designs and circuit simulations. Our programming environment distribution is done through a version controlled repository linked to subrepositories for its dependencies.

Another part of the developer kit is *modular components* which are pre-built software modules designed to be interoperable, reusable, and configurable, which speeds up development time and reduces complexity. An example of such components are IPKISS PCells developed for either tunable blocks or circuits; consider a circuit model of a MEMS coupler built based on specific experimental data, we can use this model to make a circuit model of a 7-cells hexagonal waveguide mesh and later simply change the model parameters to see the effects on circuit response. Another example, could be the python class abstracting the EIC board behaviour; users can use it to quickly buildup their measurement routine. Or, in layout design, we used the coupler and phase shifter layouts developed by EPFL and KTH to generate the layout of our reconfigurable circuits such as FP-PICs and switches.

The third item in the developer kit is *APIs* which are meant to provide a set of routines, protocols, and tools for building software and applications that interact with photonic hardware. These APIs abstract the complexity of direct hardware manipulation, offering a simpler and more flexible interface for users. In our work, we have used two types of APIs: *Hardware control* and *simulation* APIs. The first one offers functions to adjust and control physical devices such as laser sources or optical switches. And, the second one allows developers and users to programmatically interact with simulation tools, enabling automated simulation, test, and optimization of photonic circuits. For example, we have developed analysis units to automatically perform various tasks such as parasitics calculation for the mesh, finding shortest path in the mesh, or configuring the actual chip for the measurements.

4.6.1.3 Programming

The second layer of the framework is the *programming layer* which can be considered as high-level functional layer and has different aspects. The *user interface* (UI) provides a platform for users to input their desired circuit functions, which could range from simple switching matrices to complex signal processing tasks. This interface might include graphical tools for designing circuit layouts or scripting languages for defining more detailed behaviors. The *programming logic* converts

the high-level requirements from the UI into detailed control signals for the lower layers, effectively translating user commands into actionable configurations for the photonic circuit. *Circuit Configuration Management* is another part within the programming layer; algorithms at this layer set objectives for individual feedback loops based on the overall circuit configuration requirements. It ensures that each component's settings align with the circuit's functional goals, such as specific signal routing, wavelength filtering, or optical signal processing tasks.

4.6.1.4 Control Strategies

Situated above the driver electronics, the control strategies layer processes the digital signals from the hardware drivers and photodetectors to maintain circuit performance through feedback algorithms. In fact, feedback algorithms use input from the photodetectors to adjust the MEMS settings dynamically, ensuring that the circuit components operate optimally and maintain the desired optical properties despite environmental changes or component variances.

The BeagleBone runs TCP server code that listens for commands from a client PC via an Ethernet interface. All programming is done in Python and set up on each BeagleBone using Cloud9, a web or network-based IDE. Unlike traditional stand-alone IDEs, Cloud9 operates within a standard web browser and can be accessed from any computer on the same network as the BeagleBone. It allows programming from PCs, Macs, or Linux systems using just a browser. Cloud9, which comes pre-installed on the BeagleBone, includes various tools for coding, building, running, testing, and debugging software. After establishing the connections, the Cloud9 interface can be accessed by entering $http://<IPaddress>:3000/ide.html$ in the browser. Once activated, the TCP server code keeps the BeagleBone in a listening mode to handle incoming commands from the client PC.

4.6.2 Hardware Driver Layer (Electronics)

This foundational layer directly interacts with the physical components, such as MEMS actuators and photodetectors. It includes analog driver electronics and readout electronics which have been implemented using the BeagleBone and EIC boards as discussed before. The analog driver controls the MEMS actuators that adjust optical paths, phase shifts, and coupling within the circuits. These drivers convert digital control signals into precise analog voltages needed to operate the MEMS. And, analog readouts monitor outputs from the photodetectors integrated within the circuits. These electronics convert the analog signals from the detectors into digital data that can be processed by higher software layers.

One crucial aspect of this layer is that each component should have a software representation with its own parameters (e.g. phase shift as function of voltage). These need to be managed by the software as well.

4.6.3 Physical Layer (Photonics)

Within the integrated photonic-electronic stack, MEMS blocks and their electronic drivers represent a crucial interface layer between the physical photonic layer and the higher-level control and programming logic. They are directly responsible for the physical realization of the circuit's programmability, which allows for a vast array of photonic functionalities to be achieved on a single chip. In this layer, electrical signals are transmitted/received to/from the MEMS blocks and photodetectors via the bond pad grid that connects the PIC to the interposer.

4.7 Conclusion

In this chapter, we presented our modular design for the implemented electro-optic system, and illustrated the circuits on the MORPHIC RUN2 and RUN3 chips. We also discussed our packaging approach and reviewed the steps from wafer-scale fabrication and MEMS post-processing to advanced methods like wafer-level hermetic sealing and the use of interposers for complex circuit assemblies. These steps are vital for ensuring that the photonic devices are not only well-protected against environmental factors but also maintain high performance and reliability over time.

As discussed, the Electronic and Interface Control (EIC) board, in conjunction with the BeagleBone (single-board computer), forms a crucial component of the control electronics for large photonic MEMS circuits. In fact, the EIC board and the BeagleBone provide a comprehensive electronic framework that supports the complex requirements of controlling large-scale photonic MEMS circuits. This setup not only ensures precise control over the photonic devices but also allows for scalability and adaptability in various experimental and industrial applications.

Key design trade-offs include allocating space for overhead elements like bond-pads, fiber couplers, and MEMS cavity rims, which are necessary but consume valuable footprint. In circuit design, using single or multiple cavities for MEMS blocks poses a trade-off between complexity and interconnect limitations. Pattern density control is challenging due to the absence of dummies in MEMS cavities, requiring careful design adjustments to meet processing requirements. Additionally, grounding all waveguides helps prevent electrical crosstalk but restricts geometric

design freedom. Finally, controlling and calibrating digital-to-analog converters (DACs) introduces a trade-off between precision and system complexity, with calibration helping to mitigate errors.

The proposed multilayered software control architecture exemplifies the integration of electronics, photonics, and software engineering, highlighting the potential for significant advancements in photonic technologies through programmable platforms. Such software-controlled layers are becoming increasingly relevant in the photonics industry, where there is a growing need for versatile, low-cost, and quickly deployable optical systems.

5

Software Framework for programmable PICs

In this chapter we elaborate the software framework, which we have been developing to design and control large-scale programmable photonic chips based on the silicon photonic MEMS platform. Although such a framework can be used for a variety of programmable photonic circuits, our focus will be on those based on the MEMS-based recirculating waveguide meshes. Here, we present the core components of the framework and overview of its construction and functionality. For a more comprehensive and detailed understanding, users can access or request the framework documentations and examples.

The framework was initially developed to serve the MORPHIC project and has been reported in the project deliverables (not available for public access). After the conclusion of MORPHIC, the framework has been undergoing significant changes and updates, gradually deviating from its original structure. While the fundamental components remain almost unchanged, alterations have been made to the relationships between classes, the management of functions/methods, and the overall development pattern. Additionally, new features have been incorporated into the framework. The framework is still under development.

We first start with the framework introduction and architecture. Then the hardware communication and control are elaborated. Next, the core components

for various objectives such as visualizations, computations, and measurements are discussed with code examples. We also have included appendix A to demonstrate sample Python scripts for the measurement examples.

This work has resulted in a conference presentation [77].

5.1 Borna

The implementation of a Field Programmable Photonic Integrated Circuit (FP-PIC) requires a multi-layer photonic system and a robust software framework to address users' needs, from design and simulations to test and measurements. A programmable photonic system architecture usually consists of different layers of electronics, RF, and optics which enable various optical and high speed signal processing operations and calculations for a variety of applications [1]. Hence, such a system needs a software framework that manages communication between different layers of the system, facilitates design and analysis of the PIC mesh and circuit, and helps users' decision-making steps. It also should enable representing the PIC mesh (optical core of the programmable chip), its abstract/physical connectivity, configuration methods, and hardware controlling interfaces. This framework should facilitate the realization of desired functionalities like routing, filtering, and functional demonstrations which are set to be achieved. *Borna*[1] is the software framework we have developed to achieve these objectives.

Table 5.1 provides an overview of the primary layers, units, and considerations for the envisioned version of the framework, along with the status of each section at the time of writing this thesis. It's important to note that the development of the framework is a work in progress, and there are still several major steps remaining to transform it into a ready-to-use software product. Despite this ongoing process, the current version is capable of fulfilling essential tasks related to the design, optimization, and control of general-purpose FP-PICs.

5.2 Framework Architecture

To create our framework architecture, we need to define our development objectives and expected computational and experimental tasks. In fact, the framework should enable us to properly address the PIC mesh and interact with it, control the corresponding electro-optic (EO) system and hardware communications, facilitate

[1] In the Farsi language, *Borna* means young.

Development & Maintenance	Computation
Core Components ✓	Math Libraries Integration ✓
Configuration and Settings ✓	Simulation & Modeling ✓
Integration & Interoperability ⊛	Parallelization Abstractions ×
Logging & Error Handling ⊛	Data Analysis ✓
Testing & Debugging ⊛	Algorithm Implementation ✓
Protocols ✓	**Visualization**
Libraries ✓	Schematics ✓
Community & Collaboration ×	Simulation/Experimental Data ✓
Documentation	Real-time Visualization ×
Help System ×	Data Logging and Reporting ×
Educational Components ⊛	User Interface (UI) Components ⊛
Data Management	**Devises & Hardware**
Data Acquisition & Processing ✓	Device Interface ✓
Netlist Management ✓	Hardware Abstraction ✓
Experiments	Calibration and Instrumentation ⊛
Setup & Control ⊛	Hardware Emulation ⊛
Workflow Management ✓	**Design**
	Layout ✓

Table 5.1: Framework main layers, units, and considerations. ✓: Implemented, ✓: Tested not implemented, ⊛: In progress, ×: Not done.

layout design, and implement routing and configuration algorithms. In fact, Borna could be useful in different moments during the lifetime of the FP-PIC:

- High-level design (determining the size of the mesh, calculating overall losses etc.)
- Schematic design of the mesh and circuit simulations.
- Layout of the mesh for fabrication.
- Mapping the mesh to the electronic drivers.
- Hardware control: 1) Lab instrumentation interfacing with the PIC (optical switches, sources, monitors, ...) and 2) Controlling hardware of the PIC (e.g. the EIC boards).
- Testing and Calibration of the fabricated mesh.
- Configuring the mesh for a task.
- Control and feedback loops

We should clarify that this process would involve different individuals throughout the lifecycle of an FP-PIC. For instance, layout designers can utilize mesh data in their designs, while various analyses can be conducted using the mesh circuit model and block properties, such as loss and footprint. Additionally, users, even without extensive knowledge of photonic components, can easily program the mesh to achieve their desired circuit behaviors.

Figure 5.1 shows an overview of the *Borna* framework architecture where the main parts have been illustrated. As seen, it includes various tools and libraries, core Python classes (for mesh analysis, hardware abstraction, and customized tools development), netlists, main folders, protocols for internal software interfaces and classes communications, and documentation. In the following, we will dive into details and elaborate on each part.

The heart of the framework is composed of the PIC core classes by which users can create a PIC [2] and perform variety of computations and analysis on the PIC mesh. These classes are divided into *Block Aspects* (purple color) and *Mesh Aspects* (orange color), where the Blocks are either physical components (such as couplers and phase shifters) or abstract components (such as nodes and cells); for further details see Sec. 5.4. User has absolute freedom to use predefined classes or make their own customized classes and connect them to the rest of the framework using defined interfaces and protocols. Unlike the IPKISS approach where all aspects are gathered and interconnected in a PCell, we have separated these classes and created interfaces for their combination if needed. The idea behind this approach is to create light-weight classes which are easy to maintain and enable users to create complex classes based on their needs. Additionally, based on my personal experience and observation, users without programming background may have less struggle when debugging or interacting with these classes.

For hardware communications, there are five layers of Drivers, Monitors, Sources, Network, and Photonic IC and each device has been abstracted by a Python class. These classes will be used to create measurement routines and tasks. It is worth to mention that the framework employs a low-level API to establish communication with the BeagleBones. This enables the framework to transmit read and write commands to ADCs and DACs seamlessly through embedded scripts within the framework itself, eliminating the requirement for manual user intervention. For further discussion see Sec. 5.3.

As seen in Fig.5.1, there are five different tools: visualizations, computations, measurements, data management, and helpers. These tools are, in fact, Python

[2]Please note that we use blue color for the class names to distinguish between Python classes and other names. For example, PIC refers to the python class and PIC refers to the actual Photonic Integrated Circuit.

scripts including classes and functions to help users in various stages of FP-PIC design and development. Here is the task summary of each tool:

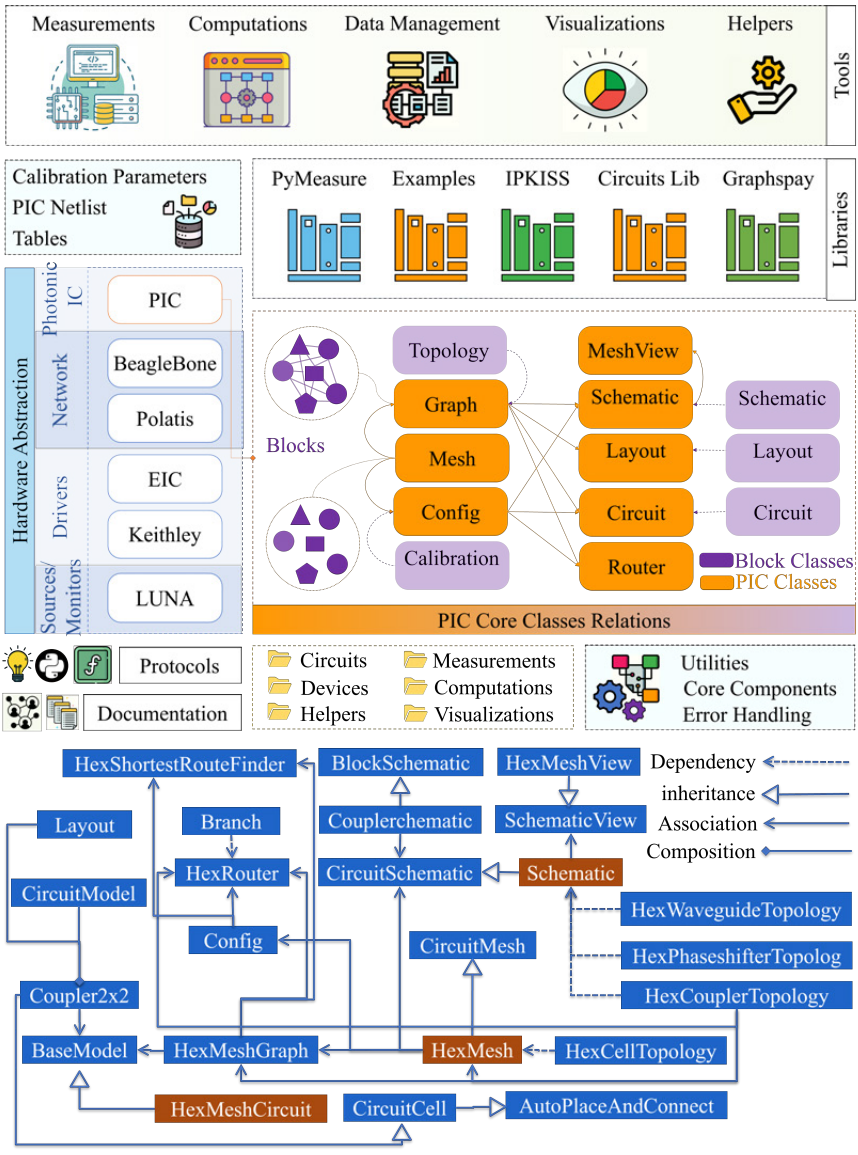


Figure 5.1: Borna framework architecture overview.

- **Visualizations:**
  - Automatic schematic generation of the defined meshes.

  - Automatic light routing within the mesh based on the selected configuration.
  - Mesh layout and graph illustration.
  - Plot assistants for parasatic and scaling analysis.
  - Plot assistants for reflection analysis.

- **Computations:**
  - Mesh circuit simulations and components models.
  - Parasitic and scaling analysis.
  - Graph-based computations for mesh routing.

- **Measurements:**
  - PIC configuration.
  - PIC characterization scenarios.
  - Hardware/Device abstraction.

- **Data Management:**
  - Data acquisition and processing.
  - Netlists.
  - Calibration parameters.

- **Helpers:**
  - Data parsing.
  - Data saving and reading.
  - Class interfaces.

The framework also includes both external and internal libraries. These libraries include tools for the layout design, predefined classes for mesh analysis and hardware communications, and examples to help users to learn how to use the framework or contribute to its development.The external libraries are:

- **PyMeasure**: A Python library designed for automating and controlling scientific instruments.
- **IPKISS libs**: IPKISS is a Python-based design framework and software platform developed by Luceda Photonics, a company specializing in photonic design automation. IPKISS is designed to facilitate the design and simulation of photonic integrated circuits (PICs).

- **GRAPHSPAY**: A code library, originally developed by my colleague X. Chen and UGENT IDlab, in which the circuit's connectivity and topology are abstracted into a graph representation.

- **Low-level API** drivers for the beaglebone software here as well, as developed by Tyndall for the MORPHIC project.

Internal libraries of the framework are *Examples* and *Circuit Libs*. In *Examples* we have covered various implementations of the framework for both computation and measurement purposes. Each circuit, such as the *hex_mesh*, has a corresponding folder where a subfolder named *lib* is present. This *lib* folder houses various pre-built elements, ranging from block/mesh schematics to circuit models. Users can leverage these libraries to initiate their analyses or measurements promptly.

In the next sections, will dive into details and elaborate each part of the framework.

5.3 Hardware and interfaces

As detailed in Chapter 4, the electronic components consist of electronic interfacing and control (EIC) boards, in conjunction with their corresponding BeagleBones (BBs). Each EIC board establishes connections with the interconnect printed circuit board (PCB) via flexible cables with 40 pins (32 of which are allocated for driving or readout signals). Additionally, these boards connect to a BeagleBone single-board computer (SBC) using a straightforward wire pair, facilitating communication via the serial peripheral interface (SPI) protocol. All BeagleBones are interconnected through an Ethernet switch, allowing remote user access for read and write commands. On the optical side, a 72-fiber array links the individual grating couplers of the Photonic Integrated Circuit (PIC) to the external world.

This packaged chip system needs to be connected to a characterization setup. The optical fibers are connected to a 32×32 Polatis switch, which is then connected to the optical sources and monitors. The connectivity of the Polatis switch can also be set by the user through the network switch. On the other side, RF sources and monitors are connected to high-speed connectors of the PCB interconnect using coax cables and to the switch network using LAN connections. Since the switch network is accessible through the internet, users can remotely control the entire EO-system.

In this section, we first elaborate the device abstraction and then dive into EIC boards controlling schemes.

5.3.1 Device Abstraction

For hardware communication with framework, all hardware components, including the PIC itself, have been abstracted through Python classes (Fig. 5.1). The low-level drivers are organized into four distinct layers: drivers, monitors, sources, and network. A primary aim of *Borna* is to streamline communication between the low-level hardware API and the PIC. It simplifies the process for users who wish to employ pre-defined measurement devices, such as a *Luna* OVA or a Polatis Switch, or integrate their custom hardware. The following script provides an example of how to define a PIC and other system components: users can effortlessly create object instances of these components and configure their respective parameters. Using these framework components they can create their own customized measurement tools, workflows, and scenarios similar to those that already have been included in the framework.

```
Code Example 1

# Init PIC class
my_pic = RUN2FPPIC()  #FP-PIC design fabricated on MORPHIC
    RUN2

# Init Electronics
my_eic = EIC(params=...)

# Measurement Instrument: Optical Switch
o_switch = Polatis()

# Measurement Instrument: Optical Sources
luna = Luna_OVA(name="luna_ova",
                host=9, timeout=2,
                connection_attemps=5)

laser_source = LaserSource()
```

As a programmer, our primary goal is to configure a mesh in order to implement a desired circuit function and subsequently control it via the low-level electronics interface. Achieving this goal involves several distinct steps:

- Specifying the states of the tunable blocks (phase shifters and couplers).
- Visual inspection of the mesh using its schematic to confirm the successful realization of the intended circuits.
- Performing circuit simulation to verify the mesh's response, and making adjustments to couplers or phase shifters if needed.

- Conversion of the selected couplings and phase shifts into voltage values to actuate tunable blocks through DAC channels.

- Comprehensive monitoring of all optical and electronic signals returning from the circuit.

- Driving adjustments, either manually or through automated processes, based on the feedback obtained from the circuit.

5.3.2 Low-level Hardware Programming Interfaces

As explained in Chapter 4, we are using the MORPHIC electronic interface and control (EIC) boards [3] to actuate MEMS couplers and phase shifters on the PIC and to read out on-chip photodiodes. And, the BeagleBone establishes communication with both the DACs and ADC on an EIC board through the standard Serial Peripheral Interface (SPI) digital communication protocol. Within the BeagleBone, a single SPI controller device interfaces with three worker devices through a Device Selector switch. To optimize the usage of SPI communication buses and minimize the number of worker devices, a single 8-channel ADC chip is shared among the 32 photodiode readouts, facilitated by an Analog Switch. The BeagleBone generates the necessary control signals by utilizing its general-purpose input/output (GPIO) pins to ensure the correct selection of the SPI worker device and the PD channels within the EIC board.

Aditionally, a stand-alone Python application is deployed on each BeagleBone, establishing communication with *Borna* via an Ethernet interface. This program is executed locally on each BeagleBone immediately upon system power-up. It operates the TCP/IP server within the BeagleBone, continuously monitoring for and responding to requests for reading photodetector (PD) values or writing digital-to-analog converter (DAC) values. These requests originate from our higher-level software framework running on a PC connected over the network. The program on the BeagleBone deciphers the incoming commands and translates them into the appropriate SPI clock and data bit sequences (Fig. 5.2). Subsequently, it generates the requisite control signals to designate the desired worker devices (DAC1, DAC2, or ADC) and PD channels based on the addressing information contained in the TCP/IP commands received from the user.

The Python library integrated into the framework consolidates all the necessary methods and properties for users to interact with the hardware of an EIC board. When users invoke the EICClient within *Borna*, the framework generates TCP/IP commands addressed to the server program running on each of the BeagleBone

[3] Developed by Tyndall National Institute during the MORPHIC project.

controllers. For every EIC board, users can create an instance of an EIC class object by specifying the IP address of the respective BeagleBone controller. Additionally, users can supply predefined calibration data for the DAC and ADC chips found in that particular EIC board.

The hybrid block diagram illustrated in Fig. 5.2 illustrates the control flow of the EIC boards, from the high-level software layer down to the hardware functional components. Through the mesh schematic interface, users can set their mesh configuration by updating Config with the `add_config_using_coords()`. This function allows users to input a `config_dict` that includes coupling (κ) and phase shift (ϕ) values for the couplers and phase shifters. Additionally, EIC classes should also be initialized with IP addresses and DACs/ADCs calibration data.

Next, the coupling (κ) and phase shift (ϕ) values will be converted to the actuation voltages for the MEMS devices through the internal methods of Config. Subsequently, the desired voltage values are converted into digital representations, and a `Write DAC` operation is created accordingly. In the case of photodiodes readout data, methods within the EICClient use pre-established calibration data to convert digital values into voltage or optical power measurements that are meaningful to the users. For each write or read function call, appropriate TCP/IP commands are constructed internally and sent over the TCP/IP Client interface to the BeagleBone where the already active Server program described previously handles the requests and interacts with the EIC board. EICClient is a wrapper around the API created by Tyndall to interface with the EIC board. The software running on the BB was also written by Tyndall.

It is worth to mention that having each EIC being self-contained with its respective BeagleBone (BB) centralizes all decision-making and control processes at the highest level, executed on the PC using Python. Although this is a flexible approach, it does impose constraints on the configuration speed for setting and retrieving the state of the photonic chip. However, given that the BeagleBone functions as a miniature computer, there is potential to shift logic and control routines closer to the EIC board. In terms of software control for the EIC board, enhancements in timing can be achieved by substituting the BeagleBone with an FPGA directly integrated into the EIC board. This change would eliminate the need for the SPI interface and offer a larger number of input-output pins for direct interfacing with all devices. Consequently, it would remove the necessity for intermediary decoders and switches within the EIC board design.

Based on the specs of our electronic hardware architecture, we can estimate the time taken by the low-level hardware interfaces and the software blocks to write the DACs and read the PD channels from the user's PC. Users can use ConfigTime to calculate the required time to implement a specific configuration or switch

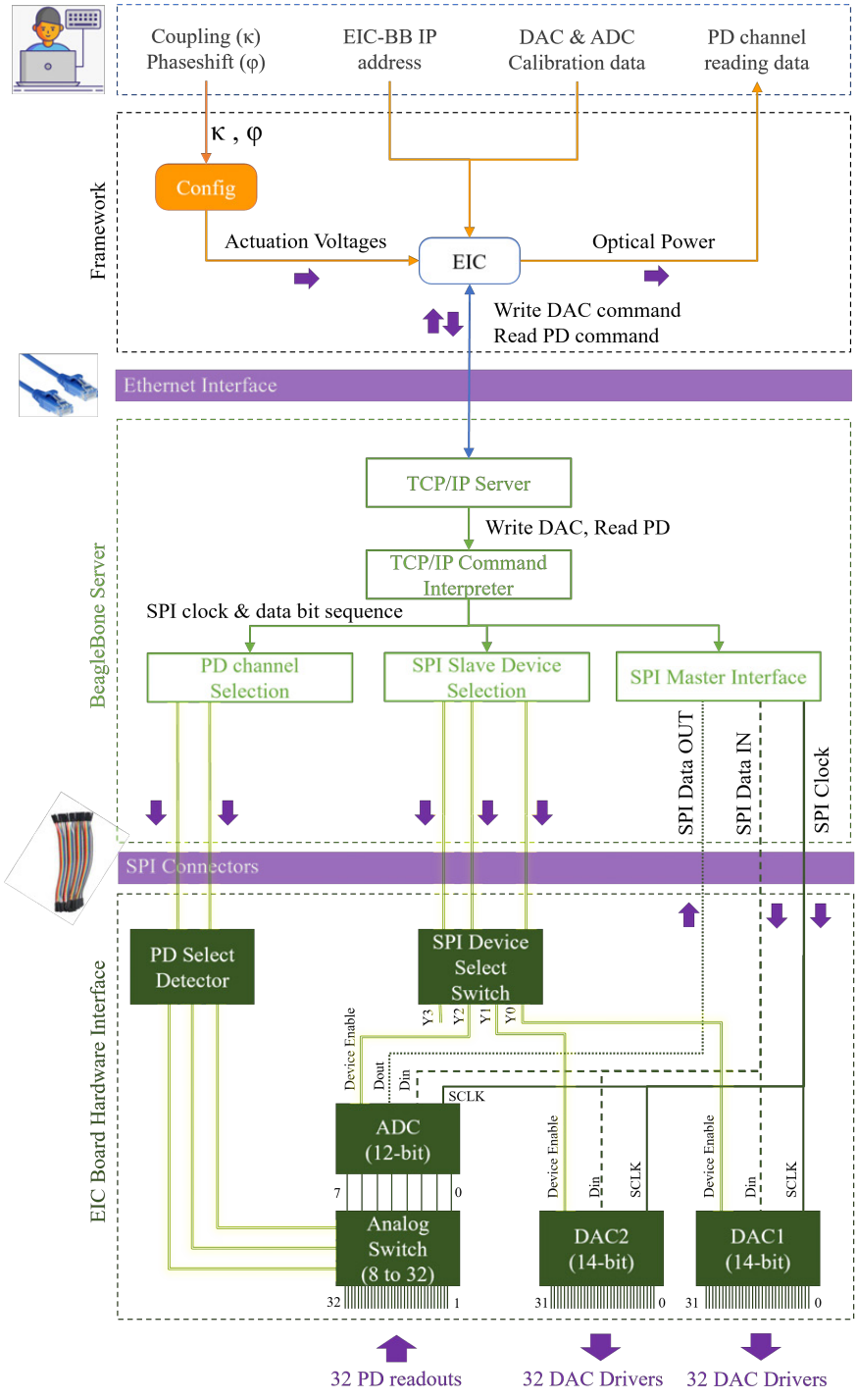


Figure 5.2: EIC Client Class incorporated into Borna, functional block diagram of BeagleBone Server Class, and hardware interfaces and functional blocks in the EIC Board.

between two subsequent configurations. For our calculations, I used the following timing parameters obtained by Tyndall measurements and wrote corresponding computation scripts added as ConfigTime to the framework:

- t_{eic}: EIC board initialization (70ms)
- t_{DAC}: Setting a single DAC channel (7ms)
- t_{DAC_bulk}: Bulk setting of 32 DAC channels (18ms)
- t_{PD}: Reading a single PD channel (8ms)
- t_{PD_bulk}: Bulk read of 32 PD channels (30ms)
- $t_{pv\_conversion}$ photocurrent to voltage conversion (100ms, usually $\geq$100ms)
- t_{EIC_PC} propagation delay from the EIC board to the PIC (1ms)

```
Code Example 2
See Code Example 3 for the mesh configuration.

my_mesh = ...
config_dict_1 = {...}
config_dict_2 = {...}

# init ConfigTime with your mesh
config_time = ConfigTime(mesh=my_mesh)

# approximate time to implement a specific configuration
t_config = config_time.get_config_time(
                                    config=my_config_1)

# approximate time of changing configurations
t_switch = my_config_time.get_configs_switching_time(
                                    config1=my_config_1,
                                    config2=my_config_2)
```

Note that Bulk command execution (Bulk setting) for DAC or PD channels refers to the process of sending or receiving data for multiple channels in a single, efficient transaction rather than executing individual commands for each channel sequentially. In fact, if the method assumes all DAC channels are being set individually, it would multiply 7 ms by the number of DAC channels (e.g., 32), which would result in a much longer execution time (224 ms) compared to the bulk setting method, which only takes 18 ms for the same 32 channels.

As a demonstration of ConfigTime, we calculated the time required to change the state of all couplers in various square-shaped meshes of different sizes. The

results are displayed in Fig. 5.3. As shown, a $(10, 10)$ mesh with 381 couplers takes approximately $0.2\,\mathrm{s}$ to be fully reconfigured.

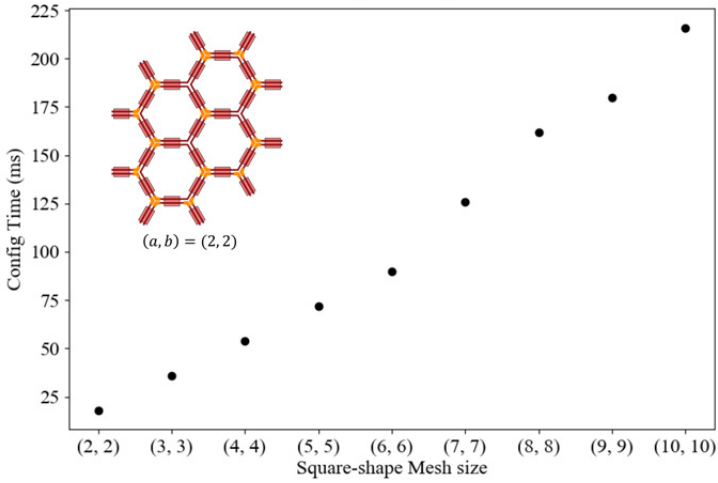


Figure 5.3: Full-mesh configuration time of the square-shape meshes of different sizes.

5.4 PIC core classes

5.4.1 The PIC class

PIC[4] is a Python class (Fig. 5.1) that can be constructed by the mesh data (configuration parameters, connectivity data, ...) and contains visualization units, computation units, and representations for IPKISS libraries (as core of circuit simulation and layout design). Users have the flexibility to choose from three options: they can opt for the ready-made PIC-based classes (for example: RUN2FPPIC or CROSSBAR4x4), create their own customized PIC class by making use of the constituent classes within the PIC such as Mesh, Config, and so on, or utilize the PIC as a super class and inherit its attributes and methods. Same scenarios exist for other classes, in fact I have tried to create light-weight and easy-to-manage classes.

PIC-based classes are those which have been developed for the MORPHIC demonstrators such as FPPIC and Switch circuits. They use PIC as their parent and have been customized based on the fabricated PICs. For example, RUN2FPPIC and

[4]It should be noted that each circuit has its own PIC class which is inherited from PIC. For example, CROSSBAR4x4 is the PIC class reperesenting the crossbar4x4 switch circuit. However, to keep the generality of the discussions we will use PIC instead of actual circuits' PIC class. Similar approach is used for the main classes shown in Fig.5.1 (ex: Mesh, Circuit,...).

BENES4x4 are ready-to-use classes in the framework to perform measurements and characterization. Similarly, users can build their own customized PIC classes by defining mesh information, netlists, and other required data. Another approach is *not using* PIC. In fact, users can perform similar tasks by using the constituting classes separately and creating their own programming flows and routines.

In the following sections, we will discuss constructing classes of the PIC (Fig.5.1) and how they are working in practice.

5.4.2 Mesh, Graph, and Config

The core of the data structure at the heart of the PIC consists of Mesh and Graph. They contain all the connectivity data, and other classes will use these for configurations, calculations, and visualizations. With the Mesh functionality, users have the capability to implement a specific topology, such as a hexagonal mesh. Subsequently, this class can be employed within the Graph to incorporate additional connectivity data. This supplementary data can encompass a wide range of information, spanning from circuit-configuration to layout-related details. For instance, to generate a circuit layout, it is essential to know the connectivity or links of each port in every component within the circuit. By utilizing graph data, we can automatically generate all the necessary links (two sets of tuples containing the connected ports of two components) for use in the `_default_links()` method of IPKISS. Similarly, for circuit simulations where specific configurations need to be implemented using auto-routing features, the implemented graph algorithms can be initialized and benefit from Graph data. This flexibility allows users to enrich their mesh representation with the necessary information to support various aspects of their design, whether they are focusing on circuitry, schematics, or other considerations.

In general, a mesh is a collection of two or more interconnected blocks, and it defines various relations between them. In the *Borna* framework, two types of blocks exist: actual (physical) blocks and abstract blocks, and each block has its own unique *id* and *coords*[5]. Actual blocks are couplers, phase shifters, and waveguides, while abstract blocks are nodes and cells. In fact, abstract blocks represent different arrangements (grouping) of the physical blocks and they can be useful for graph-based calculations and circuit simulations as will be discussed in the next sections. For example, a node can consist of 3 couplers and 3 phase shifters. Figure 5.4 shows such blocks in a hexagonal mesh schematic.

Mesh contains *ids* and *coords* of the existing blocks (both physical and abstract),

[5]Coordinates based on the defined topology.

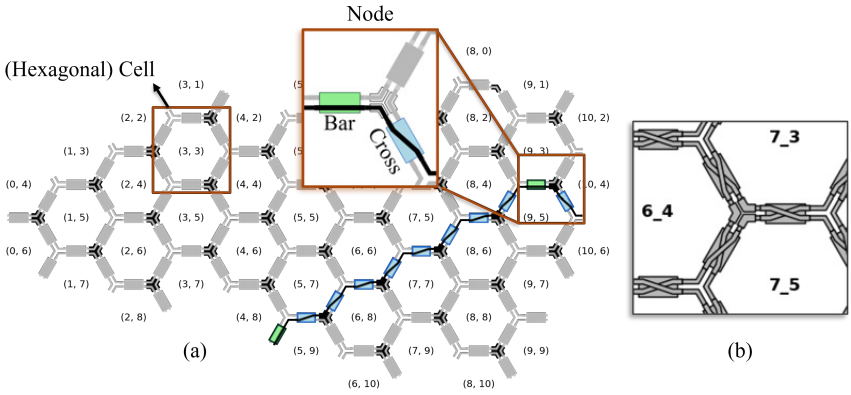


Figure 5.4: a) Schematic of a hexagonal mesh and its blocks (nodes, cell, . . .) where couplers and phase shifters have simple schematic b) a mesh node with more detailed schematic of the couplers and phase shifters.

and its built-in functions allow to easily customize it by adding/removing physical blocks. On the other side, Graph uses Mesh to create dictionaries of neighbors *ids*/*coords* for each block based on the decided topology and its relative mathematical relations. Graph was originally created to serve routing purposes; however, with a bit of modification, we made it usable by other classes such as Schematic and Circuit. For example, for circuit simulations, connectivity data of Graph will be used to form the S-matrix of the mesh circuit.

Mesh blocks can have different *aspects* of layout, schematic, circuit, graph, and calibration. In fact, when we are constructing a mesh using blocks, we should also define these aspects of the blocks which have their own related Python classes, and similar to PIC can either be used to create a Block or implemented separately. During the framework development we noticed that it is more efficient to maintain the *aspects* classes separate and use them separately to create functional classes composing PIC.

Figure 5.1 shows the flow and relation between the PIC, Mesh, and the Block objects. We first need to define blocks with their different aspects, such as their circuit model, schematic, graph representation (which is based on the mesh topology). Then, we define a mesh based on the mathematical/geometry relation of a certain topology and use the corresponding *ids* of the blocks. Below is an example to show how we can define a mesh (shown in Fig. 5.4) with hexagonal topology. As seen, we pass a list containing coordinates of the cells, phase shifters arrangement in the unit cell, and whether we like to add whiskers to the mesh or not.

```
Code Example 3

# Define the cells coords based on the topology
cells_coords = [
    (1, 5), (2, 4), (2, 6), (3, 3), (3, 5), (4, 4), (4, 6),
    (5, 3), (5, 5), (5, 7), (6, 2), (6, 4), (6, 6), (6, 8),
    (7, 3), (7, 5), (7, 7),(8, 2), (8, 4), (8, 6), (8, 8),
    (9, 3), (9, 5)]

# Each hexagonal cell can have up to 6 phase shifters
pss_in_cell = [2, 4, 6]

# Define the base mesh
mesh = HexMesh(
    add_whiskers=True,
    pss_in_cell=pss_in_cell,
    cells_coords=cells_coords)

# phase shifters modification
mesh.add_phase shifters_before_whiskers(
    apply_pss_arrangement=True)

# We can only use 'wgs', 'cps', and 'pss' as keys of
    dictionaries
added_blocks = {
    'pss': [(7, 1, 1), (6, 0, 5)]}
removed_blocks = {
    'cps': [(7, 1, 6), (9, 1, 2), (5, 1, 6), (4, 2, 6),
            (2, 2, 6), (1, 3, 6), (0, 4, 6), (1, 7, 5),
            (3, 7, 5), (5, 9, 5), (7, 9, 5), (10, 4, 4),
            (10, 2, 4)],
    'pss': [(4, 2, 4), (8, 0, 4), (9, 1, 2)]}

# Customize mesh with added/removed blocks
mesh.update_blocks(removed_blocks=removed_blocks,
                   added_blocks=added_blocks)
```

The *Borna* framework automatically generates all the required $ids/coords$ for the physical blocks (ex: couplers, phase shifters, ...) and abstract blocks (ex: nodes,...) which can be accessed using $mesh.blocks\_id$ or $mesh.blocks\_coords$, where $blocks\_id$ and $blocks\_coords$ are dictionaries. Next, we instantiate Graph using Mesh to generate all the connectivities and neighboring information:

```
Code Example 4

graph = HexMeshGraph(mesh=mesh)
```

Mesh also features two key methods: `update_block(block_id, **kwargs)`

and `save_data()`. The former enables the recording of comprehensive information pertaining to each block, including blocks aspects data, fabrication details, and calibration data specific to the block manufactured on a specific chip. The second method can be called to save all blocks data as YAML file where the file name is the same as *block_ids*.

```
Code Example 5

my_mesh.update_block('cp_3_3_3_5',
                     is_collapsed=False,
                     loss=0.5 # dB
                     schematic_name =
                         'CouplerSchematicA002',
                     circuit_name='RUN2FPPIC',
                     process_type='unprocessed'
                     )
my_mesh.save_data()
```

Another key class of the framework is Config. With Config, we can put the mesh in a desirable default state, and then set coupling (κ) and phaseshift (ϕ) values of the couplers and phase shifters. It supports both constant and wavelength-dependent values κ and ϕ. In addition, users can use their circuit models or fitted-data based on the experimental measurements of the fabricated couplers and phase shifters. To configure the coupler/phase shifters we can access them using their coordinates which are represented as tuples: ($cell\_x$, $cell\_y$, $side\_nr$)/($cell\_x$, $cell\_y$, $corner\_nr$). Here is an example of using Config:

```
Code Example 6

# Init Config class
config = Config(mesh=my_mesh,
                default_k=0.0, default_phi=0.0)

# Apply mesh configuration
config_dict = {'cps': {(6, 6, 1): 1.0, (6, 6, 2): 0.0,
                       (9, 7, 2): 1.0},
               'pss': {(9, 5, 1): pi/4.0}}}

config.add_config_using_coords(config_dict=config_dict)
```

5.5 Visualization and Layouts

For the visualizations, users can interact with the mesh schematic for various objectives such as checking the mesh configuration and identifying malfunctioned

blocks. They also can inspect the equivalent graph view of the mesh. Another feature of the framework is that users can convert their mesh schematic to the actual layout of the mesh for fabrication and then make their own customization.

To build a mesh schematic, we can use the schematic classes of the framework by passing the defined Mesh, Graph, Config, Router (Sec. 5.6.1), and blocks schematics as the input parameters. And, to construct a mesh, as explained above, cells' coordinates should be defined using a list of tuples, from which the mesh will be automatically generated. The resulting mesh can then be modified by adding or dropping components (couplers, phase shifters, waveguides) using their IDs. The reason for such an option is that when we design an actual layout based on the schematic mesh, we may need to change the arrangement of the circuit cells to fit it properly beside of other designs on the mask. Here is an example to show the flow of creating a Schematic (Fig. 5.5):

```
Code Example 7

# Update inputs and outputs with their name
inputs = {(10, 2, 4, 3): 'in1', (10, 2, 4, 1): 'in2',
          (10, 4, 4, 3): 'in3',
          (9, 7, 6, 2): 'in4'}

outputs = {(9, 9, 1, 4): 'out1', (6, 10, 6, 4): 'out2',
           (4, 8, 5, 3): 'out3',
           (2, 8, 6, 2): 'out4'}

# See Sec. 5.5 for the router discusion
schematic_router = SchematicHexRouter(
                graph=graph, config=config,
                starting_ports=inputs,
                paths_dict_generator_nr=1)

mesh_sch.update_inputs_outputs_dict(inputs, outputs)


# define blocks schematic template as a dictionary
blocks_sch_dict = {
        'cps': CouplerSchematicA002(),
        'pss': SharpBendphase shifterSchematicA002(),
        'wgs': SharpBendWaveguideSchematicA002()}

# instantiate schematic
mesh_sch = HexMeshSchematicA(
                mesh=mesh, config=config, graph=graph,
                router=schematic_router,
                blocks_template_dict=blocks_sch_dict,
                params=sch_params)
```

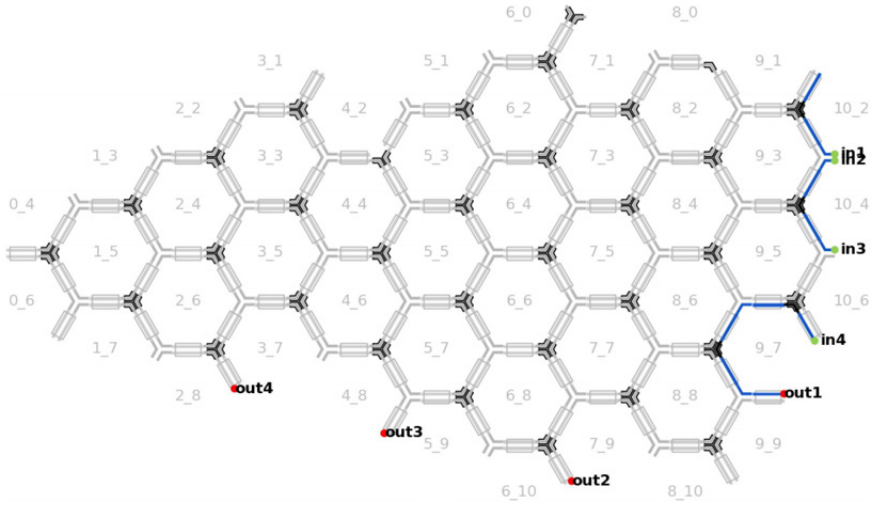


Figure 5.5: Another schematic visualization of the mesh shown in Fig. 5.4

As seen in the above script, we have used block schematics with suffix of *A002*. First we use BlockSchematic to create instances such as CouplerSchematicA; here, we have added a letter (*A*) as a suffix to represent a coupler schematics with specific shapes and geometries (compare coupler schematics of Figs. 5.4 and 5.5). Then, new objects with different styles are instantiated for the certain visualizations. These objects are stored in the framework library with a 3-digit identifier as their suffix; *002* is the selected 3-digit identifier in this example. This naming strategy makes bookkeeping process easy and scalable.

For the visualization of the schematic, we have created the MeshView, which can directly be used or be inherited by the new user-defined classes. The idea behind this class is to enable users to create their own visualizations on top of the existed Schematic. MeshView can undergo updates based on its parameters, which may include font settings for displayed text, permissions to exhibit different block types, or any other desired information. `update_params()` is a builtin function of the MeshView which takes a dictionary as its input and enables users to refresh the view with new visualizations.

```
Code Example 8

# initialize a mesh view
mesh_view = MeshView()

# Update mesh_view paramters with a dictionary
mesh_view.update_params(mv_params)

```

```
# show schematic
mesh_view.visualize(mesh_sch)
```

Using the above codes for the Schematic and MeshView, a new mesh schematic has been generated and shown in Fig. 5.5. As seen, mesh blocks have more detailed schematic compared to Fig. 5.4. This mesh is also in the Bar state which can clearly be recognized from the arms configuration of the couplers.

Figure 5.6 shows a mesh schematic of MORPHIC RUN2 FP-PIC circuit where some whiskers have been added to the mesh and some couplers have been dropped. After constructing a mesh, it can be configured to implement a circuit connectivity or define a single or multiple paths with specified input and output ports, which can be done by manually setting the coupling values of the tunable couplers using their *coords*. After defining the coupler states and input/output ports, the optical paths on the schematic can be visualized (see an example of a path in Fig. 5.6a). This is an important feature, because it helps the user to make sure that the mesh is correctly configured. Also, when we try to implement several circuits inside the mesh, path visualizations will be useful.

Another visualization feature of the framework is its graph representation, shown in Fig. 5.6b. It helps users to visually check their mesh graph which will be used later for routing and mesh configuration.

In the context of layout design, it is important to define the specific building blocks and their connectivity within the IPKISS environment. However, this process can become quite cumbersome when working on the design of a large-scale mesh, particularly if frequent alterations to its size or shape are necessary throughout the iterative design process. The *Borna* framework offers a solution by enabling users to interact with an abstract mesh and make adjustments based on their simulation results. Subsequently, the framework automates the generation of all the essential connectivity information required for the IPKISS design, streamlining the design workflow and enhancing efficiency.

To generate the layout mask layout for fabrication, we need to create new abstract blocks corresponding to the actual circuit cells on the chip (PIC cells: see Sec. 3.2.1) containing up to two nodes. This means additional connectivity and neighboring information. To extract connectivity data for the layout design, users can use `get_pic_circuit_graph()` function of the framework. Initially, they must construct their mesh graph, as previously outlined. Following this step, they can proceed to design the PIC layout using IPKISS while incorporating the generated graph data.

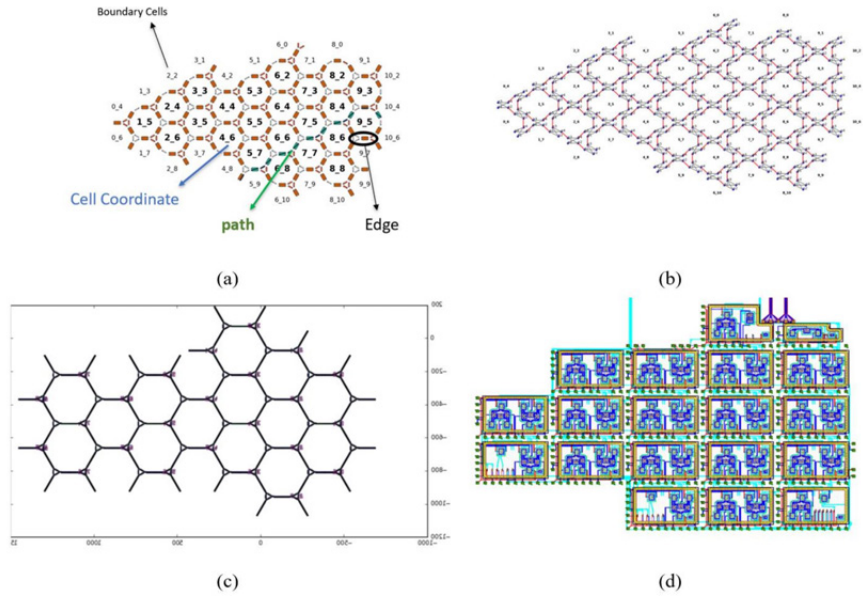


Figure 5.6: Various visualization/layouts of the customized hexagonal mesh for MORPHIC RUN2: a) Schematic, b) Graph view, c) Simplified IPKISS layout, d) Actual layout for fabrication (This specific layout was not generated in Borna, but with separate design code, because Borna did not yet exist at the time).

```
Code Example 9

# build the mesh graph as explained
mesh_graph =...

# get new graph for the pic layout design
circuit_graph = get_pic_circuit_graph(graph=mesh_graph)
```

It is worth to mention that, for customization, we should modify the Mesh, but the schematic itself will not change. For example, if we decide to change length of some waveguides or their number of bends, waveguide schematic will remain the same as before and changes will be only applied to the mask layout.

For the circuit designer, it is also necessary to translate an abstract waveguide mesh to a layout. While the abstract representation can be independent of the underlying technology, the layout depends on a particular fabrication process, along with its associated Process Design Kit (PDK).

In the context of MORPHIC, the existing iSiPP50G Process Design Kit (PDK) was expanded by incorporating functionalities tailored for the new MEMS building

blocks, supplementary post-processing layers, and packaging templates. This expansion empowers designers to engage in circuit-level design within the extended platform. It's important to note that geometric design freedom is intentionally constrained at the geometric level to enhance the reliability and yield of the standard building blocks. This PDK is defined in IPKISS.

The *Borna* framework seamlessly integrates with this PDK, combining MEMS building blocks into nodes that consist of various tunable couplers and phase shifters. Ultimately, it generates a comprehensive mesh layout. In practice, the mesh layout should be put beside other designs for the fabrication, this imposes some modifications on the boundary of the mesh. To apply such changes, users can use the general mesh as a base and then add their own customization scripts. For this mean, as mentioned in Sec. 5.5, they can extract all the circuit connectivity data using `get_pic_circuit_graph()` function of the framework.

5.6 Computations

As mentioned previously, the software framework should enable us to define a mesh, conduct a scaling analysis considering both optical and electronic constraints, speed up actual layout design process, and configure the mesh for characterizations and other measurement objectives. Therefore, we have incorporated various computational/analysis routines that can be useful in the design and characterization process. In this section, we provide detailed explanations of Python classes and their implementation within the framework.

5.6.1 PIC Configuration and Routing Algorithms

To configure an actual photonic chip, the coupling and phase shift values should be translated to the corresponding actuation voltages and then be send by the BeagleBone SBCs to the DACs on the EIC board to actuate the correct MEMS device on the PIC. The first step should be done by the user and the rest is automatically handled by the framework using the netlists, i.e. the connectivity tables that link the controller channels to the physical electric routes on the interposer on to the bondpad and actuator on the PIC.

There are two ways of mesh configurations: manually defining blocks state, or automatically calculating them using routing algorithms. For manual configuration we need to pass a dictionary (Sec. 5.5) containing coupling and phase shift values and blocks coordinates based on the mesh schematic.

To calculate the possible light paths based on defined input and output ports we can use SchematicRouter (for the hexagonal mesh: SchematicHexRouter) to route the light. This routine visualizes the flows of light in the mesh, based on the states of the blocks. Also, it does not take into account phase delays, wavelength dependencies, interference, attenuation, and the effect of partial coupling. Afterward, we can use Schematic to visualize the configured mesh and the light paths inside it. For actual actuation of the PIC we should use the `update_config()` function of the PIC as seen in the last line of the script in the following example. This final line of code will actually set the configuration in the electronic driver channels. This separation makes it possible for the user to define multiple mesh configurations and control which one is actually configured in the hardware. Here is an example for manual configuration of the PIC.

```
Code Example 10

from borna.hardware import PIC

# build the mesh, graph, config, router, and schematic as
    explained before
my_mesh = ...
my_config = ...
schematic_router = ...
mesh_schematic= ...

# Visualize the mesh to make sure the correct configuration
    has been applied
mesh_view.visualize(mesh_schematic)

# make an instance of the PIC class and then update its
    configuration, please note that this is a simplified
    code and more parameters should be used to instantiate
    a PIC class (see Appendix A for the detailed
    description)
my_pic = PIC()
my_pic.update_config(new_config)
```

For each given input, SchematicHexRouter start adding colorful paths in the couplers and phase shifters until it reaches the outputs. In some cases, defined configurations may lead to the formation of loops inside the mesh. In this cases, the router remembers ring resonators paths and only route them one time to avoid infinite repetitions.

Another built-in routing class in the framework is HexShortestPathFinder, which is a basic shortest path finder and works based on the hexagonal coordinates. This router works based on the nodes' hexagonal coordinates (Fig. 5.7). At each step, the router calculates the distance of the next two possible nodes from the end nodes

(d_{node}), and selects the closer node to the end node which is the output:

$$d_{node} = \sqrt{(x_{end} - x)^2 + (y_{end} - y)^2 + (z_{end} - z)^2} \tag{5.1}$$

where (x, y, z) are the hexagonal coordinates of the selected node. For example, as shown in Fig. 5.7, $node_B$ ($hex\_coords = (1, 0, -1)$) and $node_C$ ($hex\_coords = (0, -1, -1)$) are the possible nodes after $node_A$. Based on Eq. 5.1, d_B and d_C are $\sqrt{2}$ and $2\sqrt{2}$, respectively. Hence, the algorithm will select $node_B$ as the next node. This is a fast algorithm for the shortest path finding purposes, but the method is only valid for convex meshes. For non-convex meshes (including meshes with inner defects) it requires additional data from the mesh boundary to modify its selections accordingly.

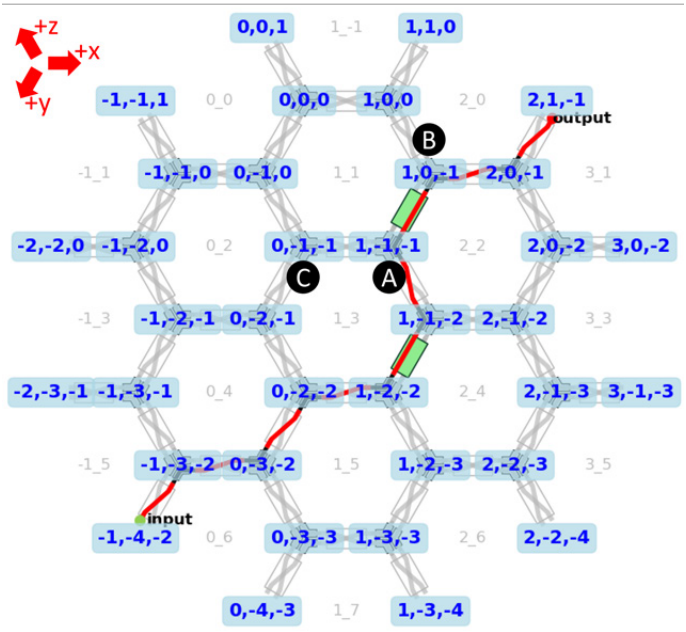


Figure 5.7: Hexagonal coordinates demonstration for the shortest path finding on a 7-cell hexagonal mesh.

It is worth to mention that although SchematicHexRouter uses Config, it does not change it. However, HexShortestPathFinder changes coupling values of the couplers to find the shortest path which means changing Config parameters. As seen in Fig. 5.7, several shortest paths can exist for a given input and output. In this case, the algorithm only selects one of them. We have used this algorithm for the loss analysis of the meshes discussed in Chapter 3.

Borna also serves as an interface for multiple graph-based libraries, facilitating the automation of routing tasks for diverse mesh configurations [3]. In this regard, users have the flexibility to employ pre-existing libraries or integrate their own custom ones. By leveraging the data structures generated by *Borna*, users can effortlessly extract graph data representing the mesh. Subsequently, they can invoke routing algorithm functions, apply the resulting configurations to the mesh, and observe the automatic synchronization of parameters across various levels, ranging from the mesh schematic to the electronic components. This streamlined process empowers users to easily transition to circuit simulations, layout design, or measurements while ensuring that all relevant parameters remain up-to-date.

Here is an example of how we can use the graph algorithms for routing:

```
Code Example 11

import new_router

my_mesh = ...
my_graph = ...

# use the mesh connectivity data stored (my_graph)
# as an interface to the new graph algorithm
new_router.init(my_graph)

# update the new graph and get new configuration data
new_router.multiple_route()
config_dict = new_router.get_config_dict()

# update the Config class with new configuration
my_config.add_config_using_coords(config_dict=config_dict)

# and finally configure the actual pic
my_pic.update_config(my_config)
```

For this, a code library was developed in which we abstract the circuit's connectivity and topology into a graph representation. To connect *Borna* to this library, we constructed a class such that each connected component in *Borna* is mapped onto the photonic graph representation with the same logical name. With this linkage, the user can choose input and output ports in the *Borna* framework, and the framework will pass on the nodes' representation of inputs and outputs to the graph library. Within the graph library, we have developed various graph algorithms for different routing tasks. A similar class is also built for the linkage in reverse, in which we translate the various data types from the routing algorithms back into the *Borna* framework. During this process, the status of the connected components gets reassigned according to the routing results. For example, the schematics and

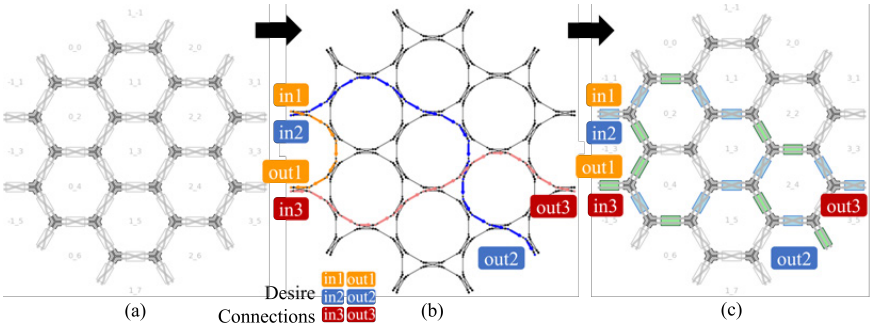


Figure 5.8: Mesh schematic converted to the graph, routed, and updated with new paths based on the configuration.

visualization will update the coupler states in the cross, bar, or partially coupling according to the routing results.

Figure 5.8 shows the conversion from a schematic to a graph representation. As seen, 3 inputs are connected to their corresponding outputs with the Shortest *combined* paths; if the orange and red paths were not needed, the blue path would have been shorter. After defining the mesh and its ports, we used graph algorithms to configure the mesh, and then we re-plotted the mesh schematic (third figure). There are 3 colors for the couplers: gray (default state of the couplers which is cross), blue (couplers with cross state involved in routing), green (couplers with bar state involved in routing). Below is the code that has been used to generate the schematics shown in Fig. 5.8.

```
Code Example 12

# Mesh -> Config -> Graph
mesh = HexMesh(
    add_whiskers=True,
    pss_in_cell=[1, 2, 3, 4, 5, 6],
    cells_coords=get_radial_mesh_cells_coords(r=1))

mesh.add_phase shifters_before_whiskers(
                apply_pss_arrangement=False)

config = Config(mesh=mesh, default_k=1.0, default_phi=0.0)

mesh_graph = HexMeshGraph(mesh=mesh)

# Schematic 1
# ------------------------------------------
blocks_sch_dict = {
        'cps': CouplerSchematicA002(),
```

```
        'pss': SharpBendphase shifterSchematicA002(),
        'wgs': SharpBendWaveguideSchematicA002()}

mesh_sch = HexMeshSchematicA(
                mesh=mesh, config=config,
                router=None, graph=mesh_graph,
                blocks_template_dict=blocks_sch_dict,
                params=sch_params)

mesh_view = MeshView()
mesh_view.update_params(mv_params)
mesh_view.visualize(mesh_sch)

# Schematic 2
# ------------------------------------------------
cxf_router.init(graph)
cxf_router.multiple_route(has_ports_paired=True)
new_config_dict = cxf_router.get_config_dict()
cxf_router.visualize()

# Schematic 3
# ------------------------------------------------
config.add_config_using_coords(config_dict=new_config_dict)
mesh_sch_2 = HexMeshSchematicA(
                mesh=mesh, config=config,
                router=None, graph=mesh_graph,
                blocks_template_dict=blocks_sch_dict,
                params=sch_params)
mesh_view = MeshView()
mesh_view.update_params(mv_params)
mesh_view.visualize(mesh_sch_2)
```

5.6.2 Circuit Simulator

In our circuit simulations, we utilize the same IPKISS toolset, which incorporates the Caphe photonic circuit simulator. These simulations can be done after configuring the mesh and assigning ports to it. Users can employ two types of circuit simulations: block-based and node-based.

For the block-based circuit simulation, each individual building block is accurately represented with its respective scattering matrix. This representation allows for the inclusion of the actual device models available in the Process Design Kit (PDK). However, for the large meshes (ex: hexagonal-shape meshes $r > 3$), the speed of circuit simulations can considerably be reduced.

For the node-based simulation, building blocks are organized into nodes, typi-

cally consisting of three tunable couplers, three phase shifters, and the connecting waveguides. In this approach, a simplified model is established for each node. This significantly reduces the overall number of elements within the circuit, leading to much improved simulation speed and efficiency.

As an example, Fig. 5.9 demonstrates the calculated transmission responses of a hexagonal mesh ($r = 1$) in *Bar* state with deviation parameters of $\sigma_\kappa = 1\%$ and $\sigma_\phi = 17^\circ$. Here, we have shown the transmission response of the mesh for three different routes. After defining the mesh as discussed and configuring it, we can use the following procedure to use the mesh circuit model.

For the simulations, we should make a *list* consisting of ports pairs where each port pair is a *tuple* constructed by the input and output ports coordinates. Each port can be addressed by a tuple: $(cell\_x, cell\_y, cp\_nr, port\_nr)$, and users can define the ports coordinates using the mesh schematic.

```
Code Example 13

circuit = CircuitSimulator(graph=my_graph,
                           config=my_config,
                           simulation_type='block-based')

my_ports_pairs = [pair_1, pair_2, ...]
circuit.calculate_transmissions(ports_pairs=my_ports_pairs,
                                wavelengths = [1.55])
```

5.6.3 Parasitics

As discussed in Chapter 2, in a mesh with a hexagonal lattice, small errors in the programmed coupling coefficients can induce chains of coupled ring resonators and perturb the circuit response with large dips in the spectrum. These imperfections will also magnify the effect of errors in phase shifters. Therefore, investigation of the programming strategies to minimize the effect of these parasitic resonances is an important step in controlling the programmable PIC based on feedback meshes.

Monte-Carlo based circuit simulation is one of the features of our framework which enables users to easily study the effect of parasitics in feedback meshes. For this mean, users can implement Parasitics in their scripts. Parasitics is in fact extension of the CircuitSimulator in which additional functions have been implemented to facilitate multi-variable monte-carlo simulations of the mesh.

```
Code Example 14

my_parasitic = Parasitics(graph=my_graph,
                          config=my_config,
```

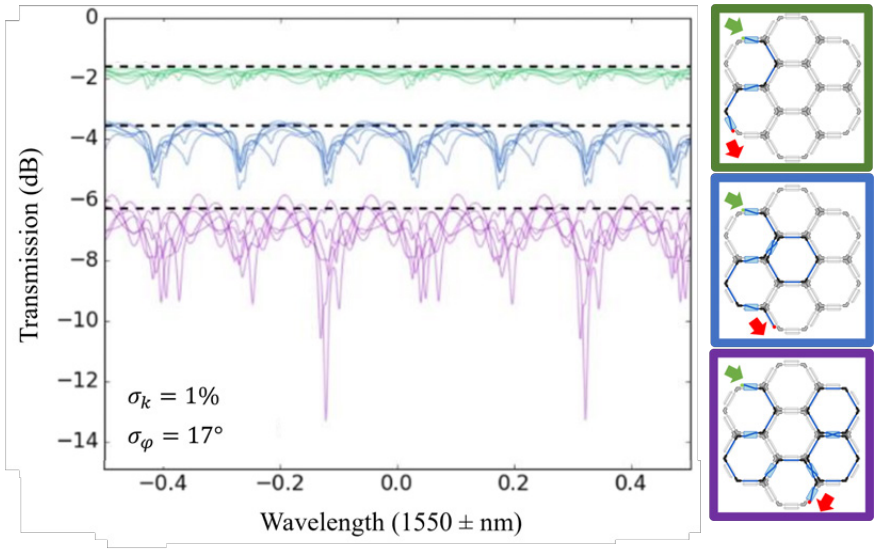


Figure 5.9: Transmission spectra of different routes programmed into a 7-cell mesh for unused couplers in Bar state. Random coupling errors with $\sigma_\kappa = 1\%$ and phase errors with $\sigma_\phi = 17°$ are introduced.

```
                        simulation_type='block-based')

parasitic.simulation_folder_dir = '...\monte_carlo_test'

# n_mc: number of monte-carlo simulaions
my_parasitic.calculate_transmission(
    ports_pairs=my_ports_pairs,
    wavelengths = [1.55],
    sigma_k=[0.001, 0.005, 0.01],
    sigma_phi=[pi/100, pi/50, pi/10],
    n_mc=100)
```

For every simulation, a directory with a distinct identifier (e.g., 00001, 00002, etc.) will be generated, and the relevant simulation parameters will be stored as a *yaml* file within the same directory. To read the transmission data from the simulation folder, the following script can be used:

```
Code Example 15

my_parasitic = Parasitics().get_transmission(
        dir= '...\monte_carlo_test',
        mc_nr=2,
        sigma_k=0.001, sigma_phi=pi/10)
```

5.6.4 Loss Analysis

One of the constraining factors in the mesh design process is the mesh loss, as discussed in Chapter 3. Utilizing the MeshLoss class, we can estimate the loss of the shortest path within the mesh for the provided ports. MeshLoss employs routing algorithms to identify the shortest path between two ports and, taking into account the specified losses of the couplers, phase shifters, and waveguides, computes the loss along the chosen path.

```
Code Example 16

mesh_loss = MeshLoss(mesh=my_mesh, router=my_router)

# to claculate loss of the shortest route between two ports
# port coordinates: (cell_x, cell_y, cp_nr, port_nr)
mesh_loss.get_shortest_path_loss(port_in=..., port_out=... )

# To calculate average path loss of the mesh
# for the all possible ports combinations (see chapter 3)
mesh_loss.get_mesh_ave_path_loss()
```

5.6.5 Statistics

Another capability of the framework is to facilitate a quick estimation of the necessary optical and electronic components and the actual layout dimensions for the specified mesh. With MeshStatistics, users can perform a scaling analysis of the mesh, as elaborated in Chapter 3. MeshStatistics can be initialized using the Mesh, and its inherent functions can be leveraged to perform computations, including the number of couplers, phase shifters, essential electrical pads on the interposer, and consequently, the count of required DAC channels. Additionally, through the use of Mapper, it enables an estimation of the layout size for the photonic integrated circuit (PIC) within the specified hexagonal mesh. This estimation is founded on the selected PIC cell architecture for the MEMS-based feedback meshes, where each PIC cell can include up to 3 couplers and 3 phase shifters. As an example, Fig. 5.10 illustrates the simplified PIC circuit based on a (9, 14) rectangular-shape mesh on the MORPHIC chip grid. As seen, this visualization quickly reveals that the selected mesh can not be fit on the grid. The Mapper was specifically created for design of the hexagonal mesh of the FP-PIC implemented in MORPHIC RUN3 and works based on the bondpad grid system defined for the chip design. To extend Mapper to more generic layouts, we should find mathematical relations describing the mapping of the nodes, based on the topology of the mesh, to PIC cells.

The following script demonstrates discussed information extraction for the

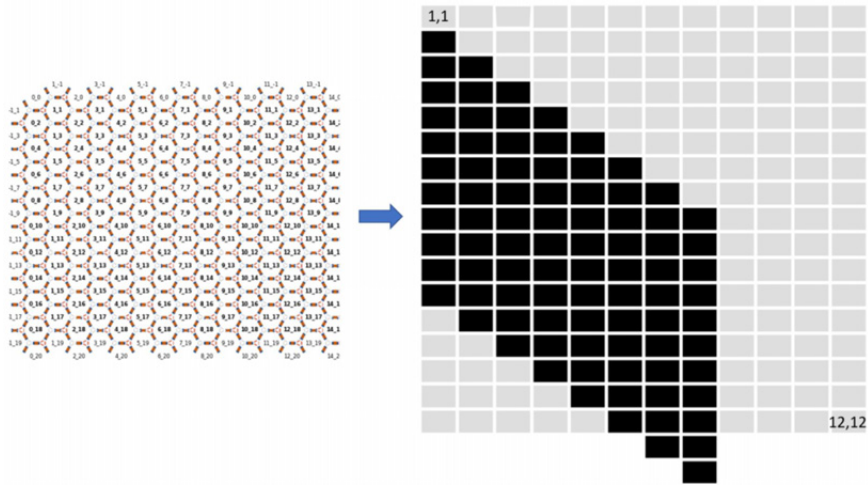


Figure 5.10: Transferring a (9,14) rectangular-shape mesh to the MORPHIC chip grid. This extends outside the available mask space.

scaling:

```
Code Example 17

# Initialization
mesh_statistics = MeshStatistics(mesh=my_mesh)

# components count
n_cps = mesh_statistics.get_n_cps()
n_pss = mesh_statistics.get_n_pss()
n_e_pads = mesh_statistics.get_n_e_pads()
n_dac_channels = mesh_statistics.get_n_dac_channels()
n_pic_cells = mesh_statistics.get_n_pic_cells()

# simplified visualization of the pic circuit
mesh_statistics.mapper.visualize()
```

5.7 Measurements

There are various MEMS-based programmable PIC measurement scenarios such as characterization and implementation of a certain or multiple circuits within the PIC mesh. However, these scenarios can be broadly classified into passive and active measurements, as discussed in Chapter 6. In passive measurements, none of the MEMS components (couplers and phase shifters) are actuated. Conversely, in active

measurements, one or more MEMS components will be actuated to implement a particular configuration in the mesh. In both cases, users are required to instantiate hardware using abstract classes, define all necessary netlists, and subsequently execute specific routines aligned with their measurement objectives.

To facilitate these purposes, we have integrated passive and active measurement units in the framework, allowing users to seamlessly execute their measurement scenarios. These units are available through PMUXXX (passive measurements) and AMUXXX (active measurements), where the *XXX* suffix can vary from 000 to 999. Users can invoke the `description` attribute of these classes to read their functionality. The development of these units has been an incremental process throughout the project, with additions made to the framework. Appendix A provide a detailed showcase of the actual implementation of these units within our *Borna*.

For efficient measurement control and scripting we have introduced two concepts of *task* and *measurement clock*. *Task* is a dictionary with keys of *'config'*, *'ports_pairs'*, and *'pds_list'*. And, it specifies which optical ports and photodiodes should be measured for the selected mesh configuration. Users should define one or several sequential tasks (as *tasks_list* which is a Python *list*), before each measurement execution. `update_task_list()` method updates the measurement unit with the given *task_list*, and `run()` method execute the measurement for each task. Then all the data measurement data will be automatically saved in the corresponding *results folder* together with the defined measurement parameters and setting. Here is an example of task definition and measurement execution:

```
Code Example 18

# define tasks
task_1 = {
    'config': {'cps': {cp_coords: coupling_1}} ,
    'ports_pairs': [('in1', 'out1'), ('in1', 'out2')],
    'pds_list': ['pd1', 'pd2']}
task_2 = {
    'config': {'cps': {cp_coords: coupling_2}} ,
    'ports_pairs': [('in1', 'out1'), ('in1', 'out2')],
    'pds_list': ['pd1', 'pd2']}
# make a kist of all tasks
task_list = [task_1, task_2]
m_unit.update_task_list(task_list)
# execute measurement
m_unit.run()
```

5.8 Data Management

In the context of a framework that integrates photonic and electronic components, effective data management in the framework ensures that information is organized, accessible, and traceable throughout the entire life-cycle of the integrated photonic-electronic system, from design and simulation to experimental validation and application-specific use. It plays a critical role in enabling collaboration, ensuring data integrity, and supporting informed decision-making. In this section, we elaborate various aspects of the data management which has been considered in our framework.

In *Borna*, we have considered several layers of data with their corresponding management tasks, which have been summarized in following:

- **Simulation Data**:
  - Store simulation input parameters and settings.
  - Save simulation results, including mesh analysis and circuit simulations.
  - Organize simulation data for traceability and future analysis.
  - Provide tools for comparing and analyzing simulation results over different scenarios.
- **Photonic and Electronic Component Data**:
  - Maintain a database or repository of component specifications.
  - Enable easy retrieval and modification of component data.
  - Associate simulation results with specific components for analysis.
  - Support versioning and revision control for component data.
- **Layout and Design Data**:
  - Store layout information for photonic circuits.
  - Track design changes and revisions.
  - Enable collaboration by allowing multiple users to work on the same design.
  - Integrate with simulation tools to link layout data with simulation results.
- **Experimental Data**:
  - Store experimental setup configurations.

- Archive raw experimental data.
- Provide tools for data analysis and comparison with simulation results.
- Support data provenance to trace experimental results back to specific configurations.

- **Netlists**:
  - PIC schematic Components
  - PIC Layout Components
  - Interposer PIC Bondpads
  - Interposer Edge Bondpads
  - Interconnect PCB Connectors Pins
  - EIC PCB Connectors Pins (DACs/ADCs)

For both computational tasks and experimental measurements, we employed dictionaries to store input and configuration parameters, saving them as YAML files. Photonic and component data are also preserved in YAML format. It's worth noting that, for a given schematic and circuit layout, multiple chips are fabricated, each possessing its unique circuit components requiring characterization. Therefore, establishing an efficient workflow for the management of this diverse data is crucial. YAML files offer user-friendly and easily readable formats, allowing simple modifications by users. Regarding layout design, IPKISS provides effective tools for data management and tracking. Utilizing PCells, users can correlate layout data with circuit models and simulations.

Within the *Borna* framework, a netlist is employed to comprehensively include all connections, varying from the abstract schematic to the channels of DACs/ADCs on the driver electronics. These netlists are progressively assembled throughout the diverse design phases involving electronic and optical components, such as the chip, interposer, PCB, and more. Notably, each element within the netlist, whether it's a DAC, ADC, phase shifter, or any other component, can be annotated with calibration data. This data is easily accessible through a lookup table. Figure 5.11 illustrates how a component within the schematic relates to the DAC/ADC pins on the EIC board. The more detailed description of the developed netlist is summarized in Table 5.2.

Another important aspect of our netlist is the Calibration Parameters. As we employ multiple EIC boards, each including several DACs and ADCs, we should have a unique address for each pin on these DACs and ADCs. To define this address, we need to identify the EIC name (e.g., '1D_01'), the DAC/ADC name (e.g., 'B'), and the specific pin number (e.g., 1). With this address in place, we

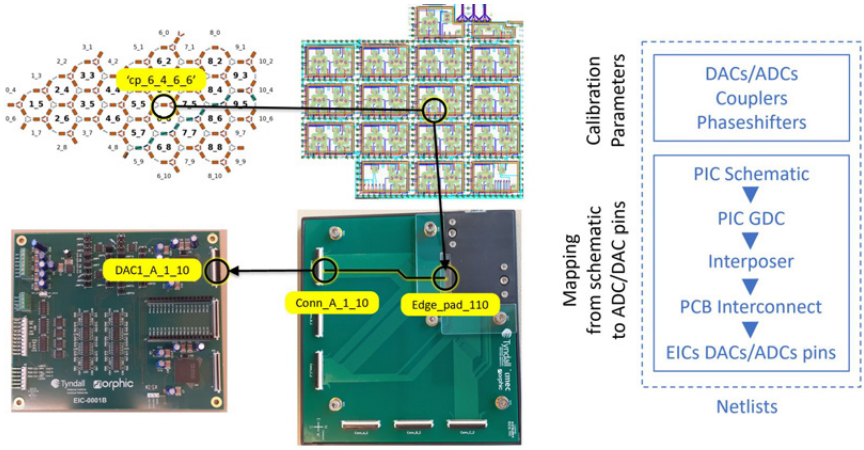


Figure 5.11: EIC Client Class incorporated into Borna, functional block diagram of BeagleBone Server Class, and hardware interfaces and functional blocks in the EIC Board.

can allocate Transimpedance Amplifier (TIA) gain values for the ADCs and offset voltage settings for the DACs. These parameters are stored as YAML file/Excel sheets and can be updated as needed. Similar situations also arise for the couplers and phase shifters, where actual coupling and phase shift values for couplers and phase shifters, determined through characterization of real chips, are stored based on component IDs.

5.9 Protocols

Another important aspect of the *Borna* development is defining protocols, data formats, and interfaces to ensure seamless communication between different classes, components, and modules within the framework. These considerations help maintaining a clear and consistent structure within the framework, enabling easy integration of new components, effective communication between existing modules, and smooth collaboration among developers working on different aspects of the framework.

The framework contains *Class Interfaces* to make sure coherency of the development. Although Python does not have a formal interface construct like some other programming languages, interfaces can be defined implicitly through conventions and abstract base classes (ABCs). We have used ABCs to define a common interface that classes should implement. This ensures that specific methods or properties are present in classes that belong to a particular category. Example of

Table 5.2: Summary of the multi-level netlist for mapping the physical components to the schematic ones. The red boxes are two netlists that should be mapped correctly using the intermediate netlists

Interface Type	Name Template	Name Example	Name Defined
MEMS Pin	GND, Vact, north	GND, Vact, ...	Component design
Coupler **phase shifter**	cp_[**#cell1**]_[**#cell2**] ps_[**#cell**]_[**#ps** in cell]	cp_6_4_6_6 ps_3_4_2	Schematic Schematic
Interposer Bondpad (PIC side)	[**#PIC_cell**]_bpd_[**#pad**]	R1C2_bpd_4	Global Layout
Interposer Bondpad (Conn. side)	edge_bpd_[**#pad**]	edge_bpd_110	Interposer
Interconnect PCB Conn. Pin	DAC1_[conn. type]_[**#PCB**]_[**#pin**]	DAC1_A_1_40	Routing PCB
EIC PCB Conn. pin	DAC1_[conn. type]_[**#PCB**]_[**#pin**]	DAC1_A_1_1	Electronics
EIC module address	[Module]:DAC1_[conn. type]_[**#PCB**]_[**#pin**]	1: DAC1_A_1_1	System assembly

such classes are Mesh, Schematic, Router, and PIC.

Another consideration is related to *data formats* for the simulation and measurement results. The common formats used in the framework are TEXT and YAML files. YAML is often considered more readable due to its clean and human-friendly syntax. It uses indentation to represent the structure, making it easier for humans to understand. TEXT files are used for storing transmission/reflection responses obtained by the measurements and, also, for simulation. The framework contains functions that help parsing data of these TEXT files based on the data analysis objectives.

5.10 Python Compatibility

The most recent version of the framework, as of the writing of this thesis, is compatible with Python 3.

5.11 Community and Integration

Borna has been used by other members of the Programmable Photonic Group for mesh simulation and analysis of programmable circuits designed and made in different projects, including the NOVA prototype currently under test in the lab.

5.12 Borna Showcase: Controlling a 7-cell FP-PIC on the NOVA chip

In this section, we demonstrate the implementation of the *Borna* to configure a heater-based 7-cell FP-PIC circuit fabricated on the NOVA chip. This circuit has been designed by my colleagues Xinag Chen and Lukas Van Iseghem, and is currently under the characterization by Yu Zhang as part of his PhD. Hence, only some simple configurations of this programmable circuit are presented here.

Figure 5.12 shows the experimental setup for the NOVA chip characterization. The setup consists of a 3-stage EIC board controlled by a micro-controller, a LUNA OVA serving as both the light source and monitor (see Chapter 6 for more details on LUNA OVA), a temperature control system (as the photonic chip includes multiple heaters), and a Polatis switch for configuring the chip's optical ports.

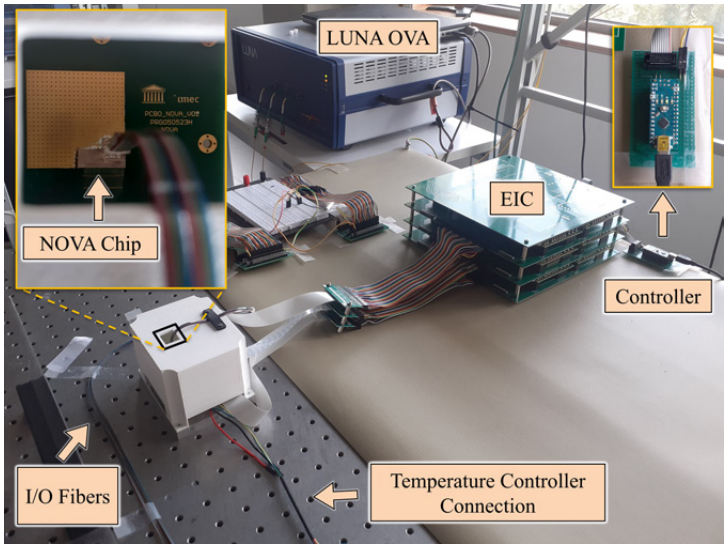


Figure 5.12: The measurement setup for characterization of the NOVA chip.

Also, the transmission measurement results for the 3 basic mesh configurations

are demonstrated in Fig. 5.13, where we have shown two simple routes, a ring resonator, and an MZI. These circuits are the ones we presented in Chapter 1 as examples of configuring a recirculating 7-cell hexagonal mesh with whiskers. Those results were obtained by the circuit simulations.

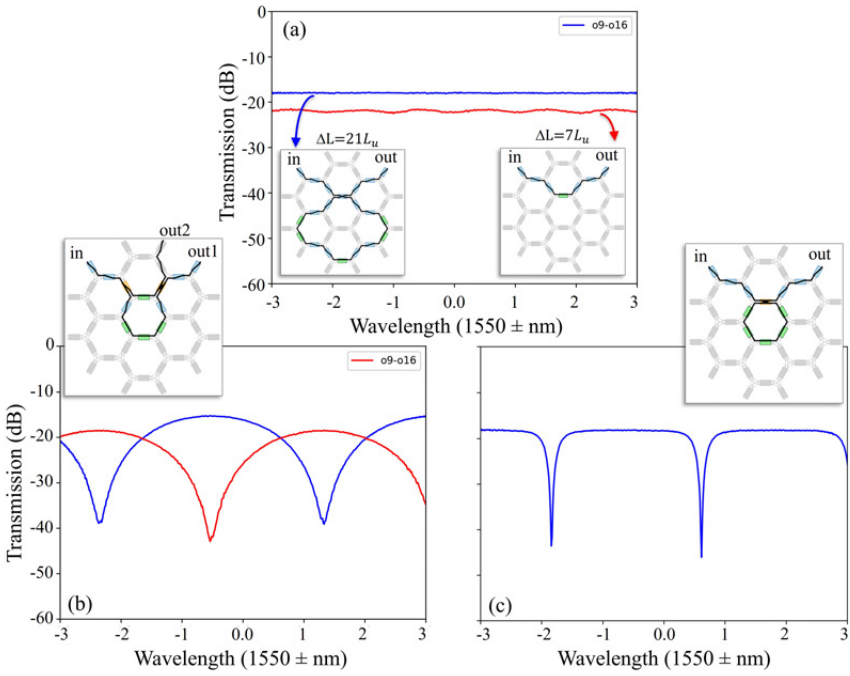


Figure 5.13: Demonstration of Borna framework to program a 7-cell FP-PIC fabricated on the NOVA chip. a) Two single routes with different lengths. b) An MZI. And, c) A ring resonator filter.

Here is the code that we used to configure the mesh and measure the optical transmission of the NOVA FP-PIC circuit for the selected configurations.

```
Code Example 19
# hardware
from borna.hardware.polatis.Polatis_switch import Polatis
from borna.hardware.luna.Luna_tcpip.Luna_OVA import Luna_OVA
from uncategorized_hardware import LTC2662 as eic
# select PIC
from borna.circuits.NOVA.pics import NOVAFPPIC
# import predefined configs
from borna.circuits.NOVA.mesh_configs import *
# analysis units
from borna.measurements import NOVAAMU001
# netlist
```

```
from borna.circuits.NOVA.netlists import polatis_map

# INIT DEVICES
# ----------------
# Optical Source: LUNA OVA
Luna = Luna_OVA(name="luna_ova", host='localhost',
                port=9, timeout=2, connection_attempts=5)
Luna.connect()
# Optical Switch: Polatis
o_switch = Polatis()
o_switch.map = polatis_map

# Electronic Driver
eic.ComOpen()
eic.reset()

# FP-PIC abstraction
pic = NOVAFPPIC()

# init measurement unit
m_unit = NOVAAMU001(luna=Luna, o_switch=o_switch,
                    pic=pic, eic=eic,
                    dir_path='results/thesis_example_001')

# define tasks
task_list = []
task_ring = {
    'config': ring_002_config_dict,
    'ports_pairs': [(9, 16)],
    'pds_list': None}
task_mazi = {
    'config': mzi_003_config_dict,
    'ports_pairs': [(9, 16), (9, 13)],
    'pds_list': None}
task_route_1 = {
    'config': route_001_config_dict,
    'ports_pairs': [(9, 16)],
    'pds_list': None}
task_route_2 = {
    'config': route_007_config_dict,
    'ports_pairs': [(9, 16)],
    'pds_list': None}

task_list.append(task_ring)
task_list.append(task_mazi)
task_list.append(task_route_1)
task_list.append(task_route_2)

m_unit.update_task_list(task_list)
```

```
62
63 # execute measurement
64 m_unit.run()
65
66 eic.ComClose()
67
68 # for visualizations
69 pic.add_config_using_coords(mzi_003_config_dict)
70 # pic.add_config_using_coords(ring_002_config_dict)
71 # pic.add_config_using_coords(route_001_config_dict)
72 # pic.add_config_using_coords(route_007_config_dict)
73 starting_ports_coords = {(-1, 1, 6, 1): 'in'}
74 pic.run_schematic_router_1(
75             starting_ports_coords=starting_ports_coords)
76 pic.view_schematic()
```

5.13 Future Development

The newly developed software framework presents a robust foundation for configuring, analyzing, and managing photonic integrated circuits (PICs). While currently functional, it stands as a work-in-progress, necessitating further enhancements and the addition of advanced features to bolster its capabilities. The framework excels in facilitating the definition of hexagonal mesh structures and switch circuits, performing scaling analyses considering optical and electronic constraints, and designing layouts for these circuits.

To further empower users, upcoming improvements aim to bolster its usability:

Integrating a more intuitive graphical user interface (GUI) for enhanced interactivity and a smoother user experience. Currently, framework is relying on Python scripts, which offers a degree of interactivity through visualization routines.

Further research can be dedicated to the exploration of graph-based routing algorithms, a subject also being addressed by GRAPHSPAY the Photonics Research Group and IDlab in the FWO project. Future developments can address both automatic single/multiple circuit implementations in the mesh. Furthermore, the development of characterization routines for automatic detection of malfunctioning actuators and the characterization/calibration of functional blocks holds significance.

To ensure precise control in the programming phase, the framework should also incorporate feedback loops (both electronic and software-based) established between the monitor photodiodes and the MEMS actuators. These programming

mechanisms are instrumental in rectifying imperfections arising from fabrication anomalies, electrical crosstalk, or fluctuations in the response curve of the actuators. This feature allows for adaptive adjustments to counter any inconsistencies, thereby refining the overall performance and functionality of the PIC. Since none of the packaged silicon photonic MEMS circuits were functioning at the time of writing, it was impossible to implement and test such feedback loops.

Additionally, the future roadmap includes a focus on refining and expanding the existing computational units within the framework to cater to more complex characterizations and versatile measurement objectives.

5.14 Conclusion

In this chapter, we delved into the intricacies of a sophisticated software framework designed for the configuration and management of MEMS-based programmable photonic integrated circuits (PICs). This comprehensive software operates across multiple layers, ranging from the localized operations executed on the BeagleBone controllers, responsible for configuring and monitoring the chip states, to the diverse layers of abstraction within the borna framework. These layers are instrumental in facilitating chip configuration and calibration, implementing automatic routing algorithms, generating diverse visualizations, conducting circuit simulations, and creating layouts for new programmable waveguide meshes.

Emphasizing modularity, the framework is designed for the integration of additional functionalities, including feedback control routines. Its foundation in Python enables the seamless integration of various existing engineering and scientific libraries, leveraging graph routines for automated routing. It has been tried to implement easy-to-manage classes with minimal coupling between them. They can be re-combined to create new classes or workflows. As a showcase of Borna application, we have been able to successfully use this framework to configure a 7-cell FP-PIC on the NOVA chip.

Overall, *Borna* stands as a promising tool in its current state, poised for further evolution and sophistication to better serve the evolving needs of photonic circuit designers and researchers. It remains a work-in-progress, continually under development and open to further enhancements.

6

Characterization of programmable PICs

Over the past chapters we discussed various mesh architectures and effect of parasitics. Circuit designs for realization of our programmable PICs were explained in details and related optical system architecture and integration were elaborated. We also discussed our software framework which can be used both for design purposes and controlling system hardware. In this chapter, we present characterization results of the fabricated and packaged demonstrators including the switches as part of MORPHIC project. To keep coherence and a smooth flow of the content presented here, I have used MORPHIC deliverables content and contributions from some of our colleagues' work during the MORPHIC project in this chapter.

As will be described in detail, we faced a large-scale collapse of the MEMS devices in all circuits, and there was little or no light going from the input fibers to the outputs. In fact, when the suspended waveguides collapse and stick to the silicon substrate, light leaks away into the substrate. Other MEMS states, for instance where two waveguides are stuck together, would still give acceptable transmission. With this observation, the testing of the demonstrators turned into a debugging operation: how widespread is the collapse? Does it occur only in some waveguides of the connected circuits, or are unconnected circuits also affected? If we identify paths that are still transmitting light, can we actuate them, or are the MEMS no longer mobile? And which process step is responsible for this phenomenon? In fact, the debugging process is in some ways more difficult than simply characterizing

working devices, as you have fewer ways to collect a meaningful signal.

To answer these questions, different techniques have been used. As the demonstrator chips are flip-chipped and sealed inside a package, we cannot visually inspect them. Even if we would destructively disassemble a demonstrator, we cannot be sure that this disassembly process is not damaging the MEMS devices. Hence, we had to find ways to probe the inside of the circuits, either on the demonstrator itself, or on proxy chips that underwent similar processing. We made extensive use of reflection measurements (OFDR), comparing the packaged demonstrators with unprocessed chips. We opened up sealed MEMS cavities of chips that went through simpler packaging flows but still showed signs of collapse, which suggested that the origin of the problem is not related specifically to the thermal processes in the assembly flow, but other mechanisms such as Electrostatic Discharge (ESD), Ultrasonic Vibrations, and Substrate Grounding which are discussed at the end of this chapter.

6.1 Measurements Overview

For our measurements we used both packaged and non-packaged chips. The packaged chips are referred as mini and full demonstrators. As discussed in Chapter 4, the full demonstrators (FDs) include a multi-layer interposer and support all the demonstrator circuits implemented on the PICs. In contrast, the mini demonstrators (MDs) include a customized single-layer interposer and support only some of the circuits on the PICs. For this work we received a full demonstrator and two mini demonstrators (the first version (MD1) uses a glass interposer and the second version (MD2) uses a Si interposer). In this chapter, we present only the results of the mini demonstrator version 2 (MD2) due to its better performance compared to MD1, even though neither version has functioning MEMS circuits.

The measurement process of the non-packaged samples includes unprocessed, etched, and sealed chips (Fig. 6.1). In unprocessed chips, MEMS devices have not been released, while in etched samples the MEMS components are released and the connecting waveguides are suspended. And, finally, sealed chips have undergone a wafer-level sealing process that allows placement of the thin caps above the MEMS cavities, which protect MEMS structures from environmental influences and for handling without the risk of damage. Measurement of the non-packaged chips helps us obtain more insight into the effect of each stage on the circuits' performance (loss, components failure, etc.).

Table 6.1 shows the summary of the measured demonstrator circuits. The packaged chips include switch circuits of: *Crossbar 4×4*, *PI-loss 4×4*, *Benes 4×4*,

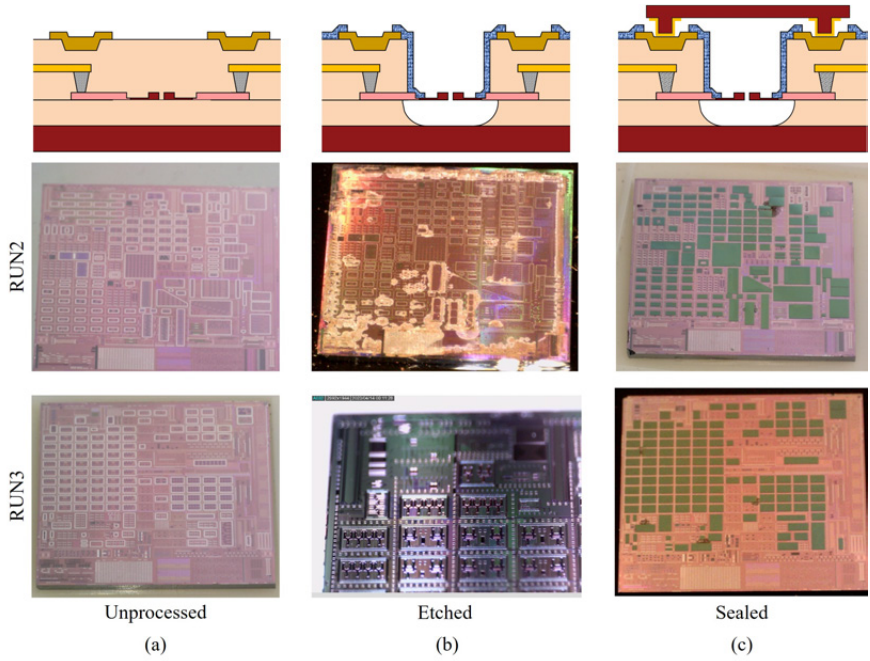


Figure 6.1: Examples of characterized MORPHIC RUN2 and RUN3 chips: a) Unprocessed, b) Etched, and c) Sealed

and *Benes 16×16*. The FP-PIC circuits are limited to the non-packaged chips, and none of them is working for both sealed and etched samples.

	RUN#	Non-Packaged			Packaged			Fiber Array
		unprocessed	etched	sealed	MD1	MD2	FD	
Crossbar 4×4	2	✓	✗	✗	✗	✗	✗	A
PI loss 4×4	2	✓	✗	✗	-	✗	✗	A
Benes 4×4	2	✓	✗	✗	-	✗	✗	A
Benes 16×16	2	✓	✗	✗	✗	✗	✗	A
24-cells FP-PIC	2	✓	✗	✗	-	-	-	B
7-cells FP-PIC (Heaters-based)	2	✓	-	-	-	-	-	B
126-cells FP-PIC	3	✗	✗	✗	-	-	-	A
Test Nodes	3	✓	✗	✗	-	-	-	-
Interposer	-	-	-	-	Glass	Silicon	Ceramic	-

Table 6.1: Summary of the characterized circuits fabricated on MORPHIC RUN2 and RUN3.

6.2 Characterization Setups

To perform our chip characterizations, four different measurement setups have been implemented:

a) Passive measurement of unprocessed/etched/sealed chips,

b) Passive measurement of the packaged chips,

c) Actuation measurement of non-packaged chips, and

d) Actuation measurement of packaged chips.

To obtain a circuit's optical response, an Optical Vector Analyzer (OVA5000) from *Luna* has been used. This tool integrates a tunable laser source and a coherent receiver with an interferometer. It collects both transmission and reflection for both fiber polarization states over a wavelength of 1525-1610 nm. This allows us to perform transmission and reflection wavelength responses (Fig. 6.2), and as the OVA also collects phase information (through its built-in fiber interferometer) we can also get the time impulse response through an inverse Fourier transform of the collected spectrum. A transmission measurement uses both fiber ports of the *Luna* OVA, while in reflection only one port is used. This latter technique is also called Optical Frequency Domain Reflectometry (OFDR). The results can be plotted as a time trace or the equivalent optical length.

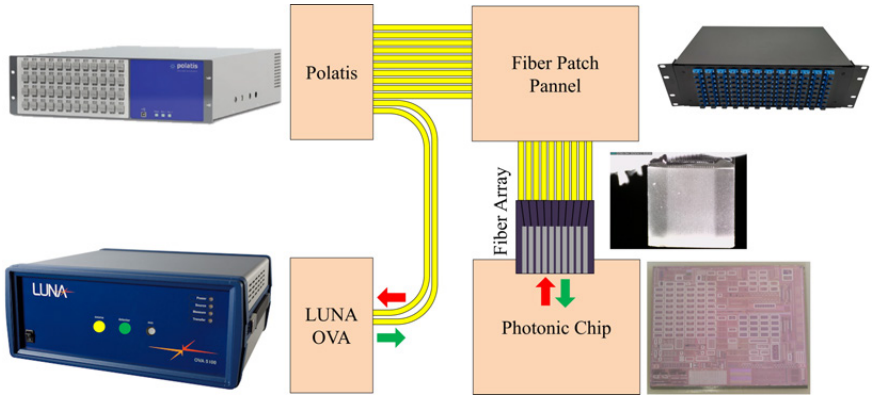


Figure 6.2: Schematic of a typical setup for transmission and reflection measurements using LUNA OVA.

Figure 6.3 shows an example of the reflection response using the *Luna* OVA. We usually see several peaks which can be attributed to the fiber connectors, the

Polatis switch, and devices on the PIC such as the grating couplers and MEMS couplers/phase shifters, waveguides crossing. The strength of the peak is proportional to the reflectivity of the element and the level of optical power that is incident on this reflector. Losses along the path are counted twice, as the reflected light needs to travel back the same way. Also, there are distributed reflections due to backscattering in waveguides. This results in a slowly decaying plateau. For optical fiber, the backscattering is so small that it corresponds to the noise floor of the measurement, but for on-chip waveguides the backscattering is much higher. The

Figure 6.3: An example of the reflection response using the Luna OVA in OFDR mode. There are several peaks which are related to the fiber connectors, the Polatis switch, and the PIC components such as the MEMS couplers/phase shifters, waveguide crossings, transitions from MEMS vacuum to the oxide cladding. Here, the red box highlights the position of the circuit peaks, and the yellow arrow shows the peak related to the connection of the fiber patch cables and UPC fiber connectors.

waveguide backreflections help us identify the peaks related to the PIC: These reflection peaks are normally positioned on a broad plateau, while peaks in the fiber setup show up as single peaks. Another feature in the reflection response is that there is always a peak located almost one meter before the first peak of the circuit (corresponding to the input grating couplers). This peak is because of the

connection of the fiber patch cables with UPC fiber connectors [1] of the packaged chips which have the length of almost one meter.

For a better understanding of the reflection measurement process, the reflection response of the Crossbar 4×4 circuit (see 3.2.7.1 for the circuit details) on an unprocessed chip when light is injected to the input of coupler 14 ($in4$) is illustrated in Fig. 6.4. The similar demonstration is used to elaborate response of the circuits presented in the next sections. As seen, the reflection response has three major peaks regions related to the input grating couplers (green), circuit components such as couplers or crossings (blue), and output grating couplers (red). It also has flat parts corresponding to the on chip waveguides. Four peaks clearly indicates that light can reach from $in4$ to $out4$ and passes through four couplers of 14, 24, 34, and 44. It should be mentioned that in all reflection plots the distances correspond to the free space light propagation.

Figure 6.4c shows the schematic of the possible reflections for the grating couplers regions on Fig. 6.4a. Ideally, we expect to see only one peak for the input grating coupler and one peak for the output grating couplers. However, our investigations show that there are two major peaks equally distanced from each other for both input and output grating couplers; these *peak pairs* are indicated by the black triangles, as markers, in Fig. 6.4a. We have also used the numbering to correlate the peaks in the reflection response shown in Fig. 6.4a with their corresponding locations in the reflections' schematic in Fig. 6.4c. One peak in the pair is associated with the grating couplers (numbers 1 and 3), while the other one corresponds to the interface between the substrate and the SOI (numbers 2 and 4). To verify our reasoning, we measured the distance between the peaks in each pair for both input and output grating couplers for most of the circuits and shunt waveguides. The measured value is almost the same for all cases and approximately equal to $2600\,\mu\mathrm{m}$. The depth and refractive index of the substrate layer are $725\,\mu\mathrm{m}$ and 3.45 which means that the distance between the two peaks in the peak pairs should be around $2501\,\mu\mathrm{m}$. This value is close to the one we measured from the reflection responses confirming our assumption.

We should note that grating couplers regions may have several or no peaks. Additional peaks in the input grating coupler region could be attributed to defects near the grating coupler. And, for the output grating, absence of the peaks could be attributed to the constructive or destructive interferences of light inside of the circuits, or the fact that collapsed MEMS structures do not allow light to reach the outputs. The assumption that peaks in the output grating couplers' region are eliminated through destructive interference can be confirmed by observing

[1] The fiber arrays purchased by Tyndall had UPC connections. In such connectors, the fiber facets don't have an angle, and therefore we observe a stronger backreflection than the APC connectors typically used in our measurement setups.

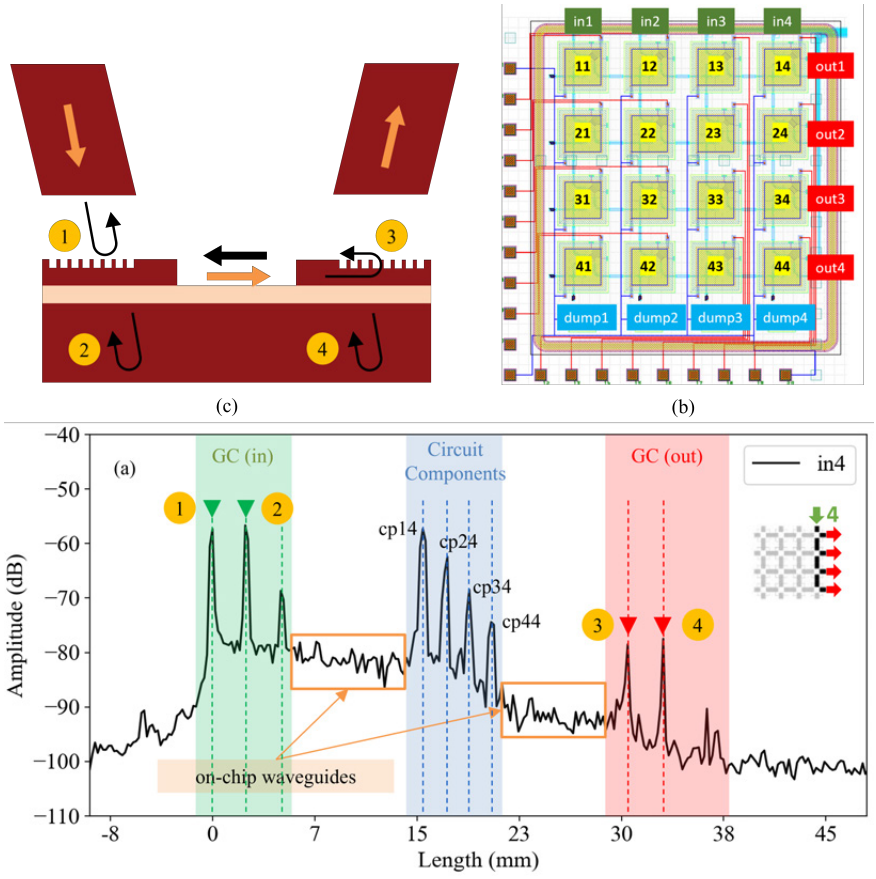


Figure 6.4: a) Reflection response of the Crossbar 4×4 *switch when light is injected to the* $in4$ *port. The response has three major peaks regions related to the: input grating couplers (GC(in),green), switches crossings (blue), and output grating couplers (GC(out) red) b) Circuit layout. c) Schematic representation of the tree possible reflections from the grating couplers.*

a scenario where light reaches the output coupler, but no peak is detected. We should note that destructive interference should show up in the time trace, because destructive interference will probably have a wavelength dependence.

Characterized demonstrator circuits have many optical input and output ports connected to a 72-fiber array. To avoid manually connecting fibers and to automate measurements, selection of fibers and the optical instruments are connected to a 32×32 Polatis switch, which can be configured either using web browser or Python scripts. This enables us to make the desired connections between OVA's source/read ports and the optical ports of the PIC circuits. Also, APC fibers are used to connect

Polatis switch to the packaged chips, the instruments, or the fiber array patch panels.

On the electronic side, a BeagleBone Black(BB) together with an Electronic Interface and Control (EIC) board sends actuation commands to the PIC; the BB and EIC board(s) are connected and SPI commands are sent over a serial connection. The EIC board is also connected to the PCB interconnect of the packaged chips using 40-pin flex connectors. Each BB-EIC module supports 64 DAC and 32 ADC channels and for large circuits where more channels are required additional modules should be used.

While the demonstrators come packaged with fiber array connectors, we also want to characterize unpackaged chips. For this, we couple in light with a 24-channel or 72-channel fiber array mounted on a PI Hexapod six-axis alignment stage with active feedback control for automatic alignment. The alignment is performed by optimizing the power transmission through a shunt waveguide on the chip between two grating couplers. This can be performed once at the beginning of a measurement, or the alignment can be actively maintained with a hardware feedback controller that actuates a piezo nanocube mounted on the hexapod.

All instruments (OVA, Switch, hexapod, EIC board, other sources, and power meters) are controlled through Python scripts running on one PC. This allows us to construct complex measurement procedures, for instance to collect all transmissions between all input and output ports of a switch circuit.

As the available fabricated demonstrators (MD1, MD2, FD) mostly contain switch circuits, the measurements that are presented here mostly concern transmission and reflection measurement, and no high-speed measurement or free-space measurements (for beamforming) have been conducted. As already mentioned, and as will become clear in the following sections, the measurements on these demonstrators do not show promising results: see Table 1 for the measured circuits, where crosses and ticks indicate failure and success of the measurements.

6.2.1 Non-Packaged Chips: Passive Measurements

As the first step, we are interested to know the response of the circuits before releasing and sealing the MEMS. This is an insightful experiment to give us an idea of how circuits could respond when they were in their initial configuration (ideally cross state but due to design considerations they are in partial coupling state close to the cross state). These unprocessed chips cannot be actuated (the MEMS are not released) and have a slightly different optical transmission because the waveguides are largely surrounded by oxide, and not by air or vacuum.

Figure 6.5 shows the setup we used for passive measurement of unprocessed

chips. All optical connections, OVA source/read, hexapod optical readout, a Syntune fixed laser, and the fiber array (with 24 or 72 fibers) are managed through the Polatis switch (Fig. 6.5d).

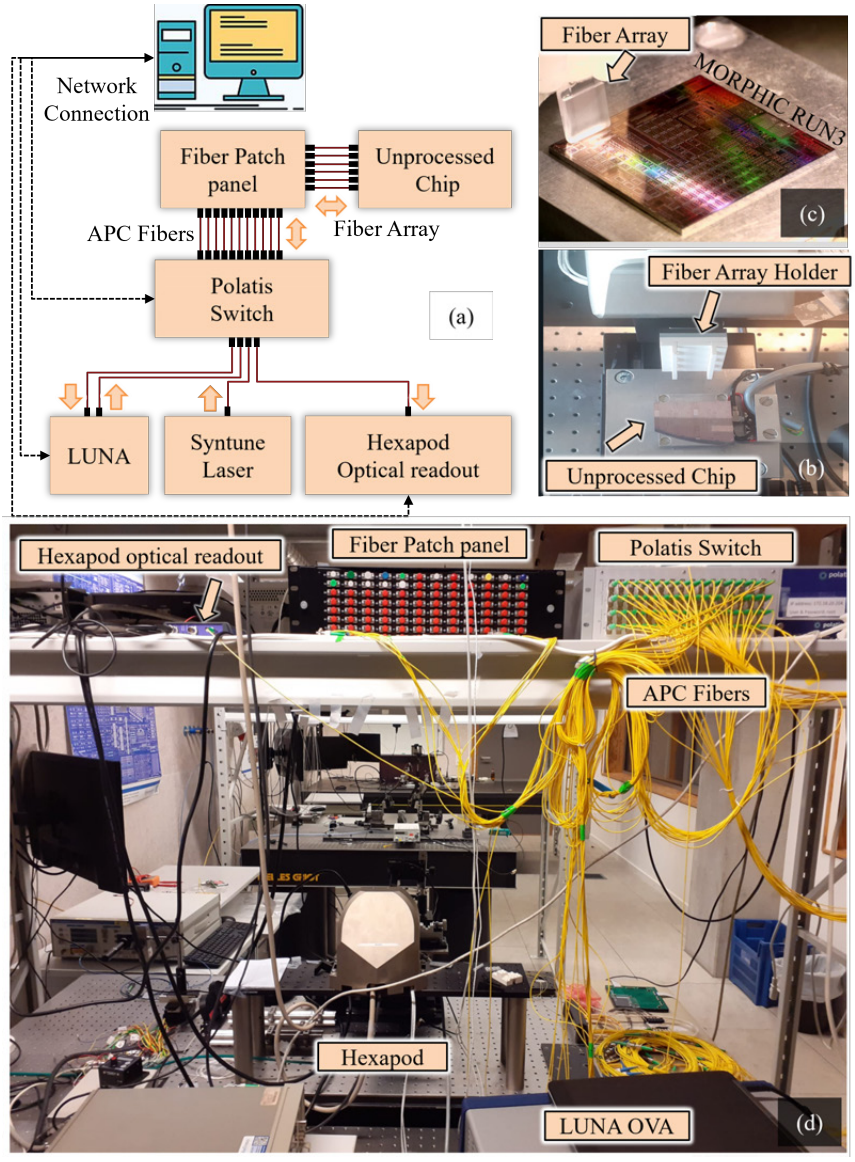


Figure 6.5: The measurement setup for unprocessed chips. a) schematic of the setup, b) front view of the hexapod and its fiber array holder, c) attached fiber array to the hexapod and unprocessed chip, d) actual measurement setup.

6.2.2 Packaged Chips: Passive Measurement

As the second step, passive measurements (no actuation of the MEMS) on the packaged demonstrator chips have been carried out to evaluate performance of the circuits after release and packaging. These measurements show the situation of the circuits when they are in their initial configuration after packaging, and before they are electrically connected. This is important to rule out that the collapse of the MEMS devices is due to the connection with the driver electronics.

Figure 6.6 shows the setup used for these measurements. Here, the *Luna* OVA and the packaged chips are connected to the *Polatis* switch using APC fibers (a patch cord is needed as the packaged demonstrators come with UPC fiber connectors). The first step is configuration of the Polatis switch using python scripts and then using the OVA for transmission and reflection measurements.

As will be discussed in the next sections, the passive measurements on the demonstrator circuits show very low transmission, as many of the MEMS devices have collapsed. When the MEMS actuators are stuck to the underlying silicon substrate, light leaks into this substrate and is lost. However, measuring these devices in reflection gives us a lot of information.

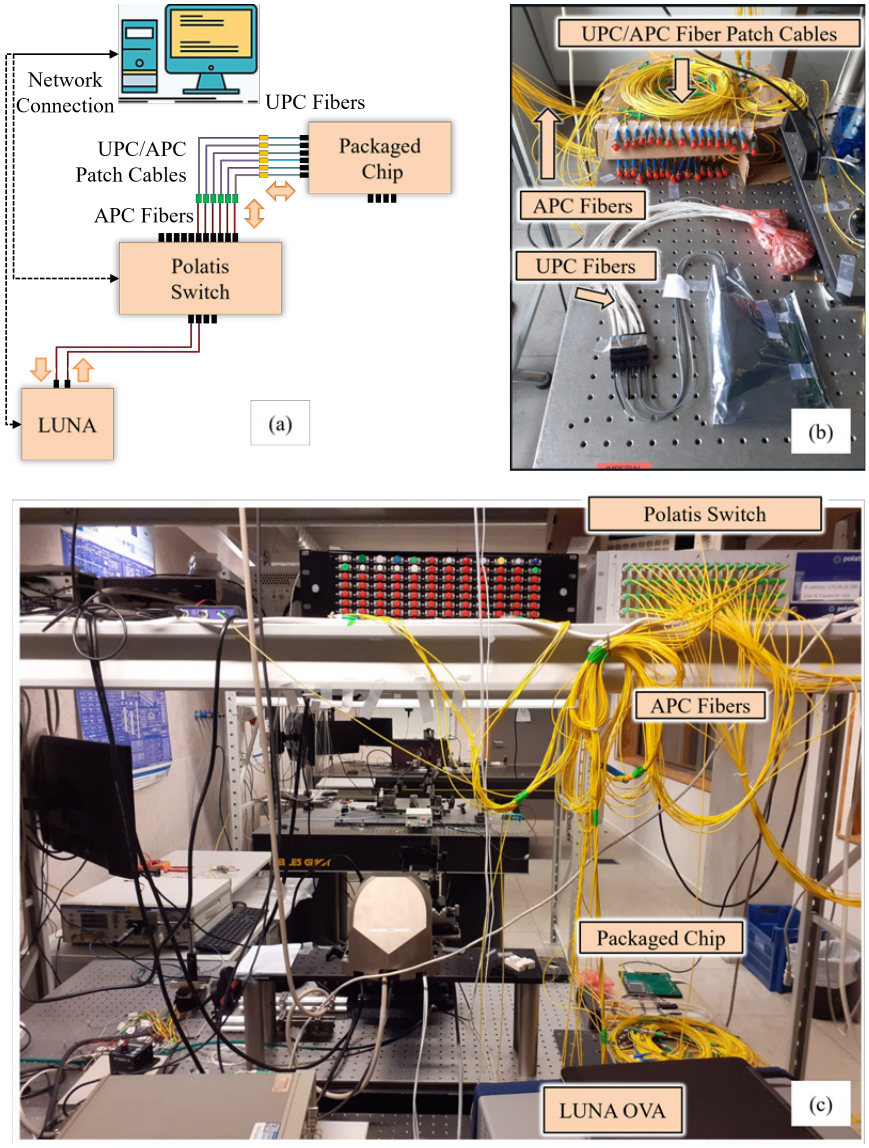


Figure 6.6: a) The schematic and b) measurement setup for passive transmission/reflection measurements of the mini/full demonstrators.

6.2.3 Packaged Chips Actuation Measurement

The last type of measurement is to measure how the circuits respond when MEMS structures are actuated. As shown in Fig. 6.7, the EIC board is connected to the full demonstrator using the 40-pin flex cables and the BeagleBone to the EIC board using SPI connection. For these measurements the *Polatis* switch is configured to set the optical inputs and outputs, then actuation voltages are sent to the PIC through BeagleBone and EIC board, and finally transmission/reflection of the circuits are measured using the OVA. Since the MEMS structures are sensitive to electrostatic discharge (ESD), this type of measurement should be conducted using a reproducible protocol. After establishing hardware connection, power sources should be turned on in the correct order, making sure all parts of the setup are connected to a common ground plane.

As will be clear when describing the different measurements, the MEMS devices did not respond well to the actuation. This can be attributed to two causes:

- Many MEMS phase shifters and couplers have collapsed during the packaging flow, which is confirmed by the passive optical transmission measurements.
- The devices that have not collapsed might have survived because they were not properly released in the first place. Especially on circuits from RUN2, the narrow gaps in the MEMS couplers and phase shifters were sometimes bridged, and this can mechanically stabilize the devices. But at the same time, this means that the devices cannot easily be moved, and they will have a very weak actuation response.

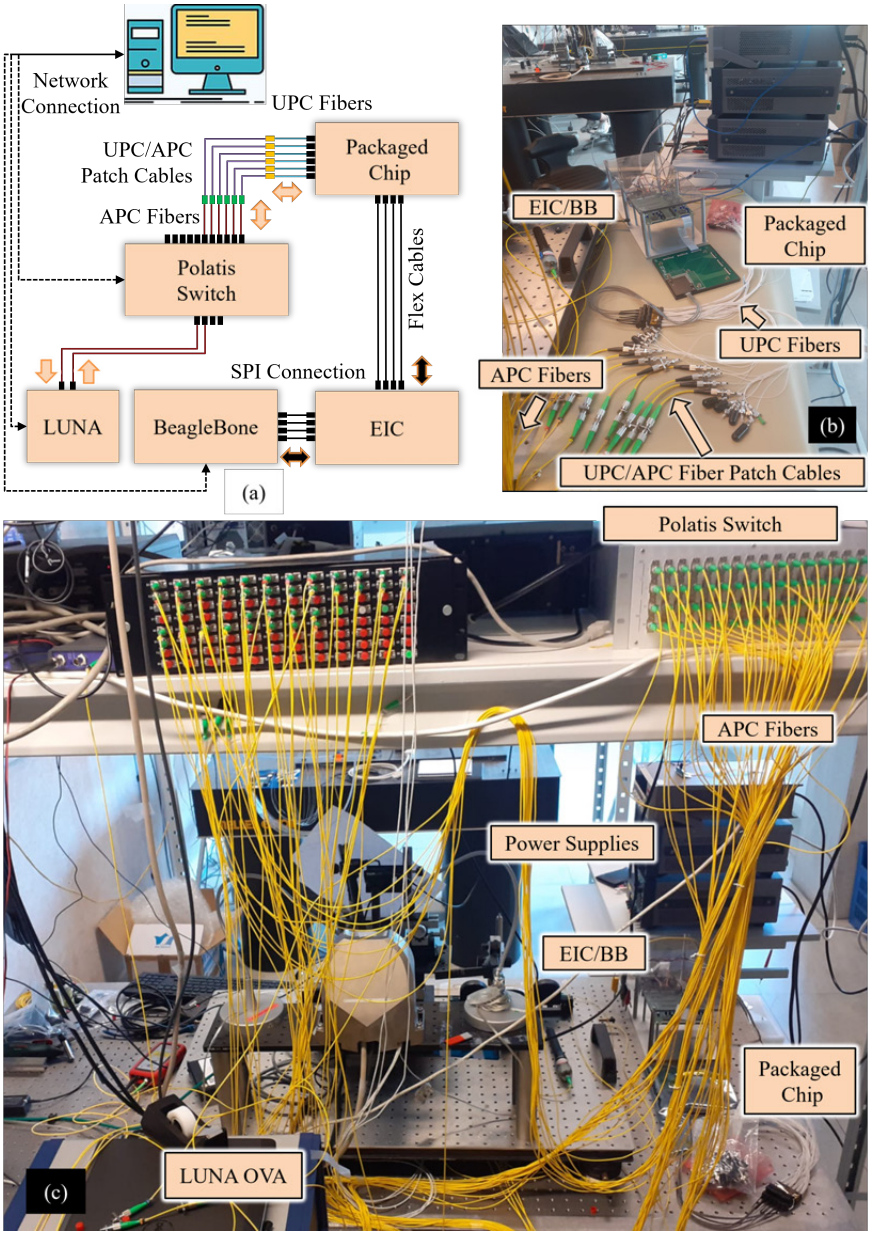


Figure 6.7: The measurement setup for actuation of the packaged chips.

6.3 Circuits Measurements

In this section, we first present the passive measurement results for the Crossbar 4×4, PI-loss 4×4, Benes 4×4, and Benes 16×16 circuits for both packaged and unprocessed chips [2]. Next, we demonstrate two actuation cases for the Crossbar 4×4 in the full demonstrator (Sec. 6.4.1) and the test nodes in one of the sealed chips from RUN3 (Sec. 6.4.2).

We should remind that, in addition to a fully packaged demonstrator, two mini demonstrators of the RUN2 switches were assembled, one based on a glass interposer (MD1), and the other one based on a silicon interposer (MD2). The first version of these mini demonstrators only contained the Crossbar 4×4 and Benes 16×16 (the other switches were not sealed), while the second one also included the PI-loss 4×4 and Benes 4×4 circuits. The first mini demonstrator (MD1) with the glass interposer suffered from bad electrical contact with the solder balls, so here our focus will be on the results of the second version.

To discuss circuits characterizations, we have devoted a subsection to each switch circuit, where we discuss the transmission measurements, reflection measurements, and Inverse Fourier analysis of the LUNA OVA obtained from the transmission measurements. We start with the unprocessed chip and then discuss the full and mini demonstrators.

Analysis of the transmissions/reflections of circuits on unprocessed chips (i.e., chips as they arrive out of the iSiPP50G process flow) is an essential step to understand the behavior of the circuits. As no underetch process has been applied, the MEMS cannot collapse in this case, and we can trace light for all the possible combinations of the inputs and outputs. The characterization of unprocessed chips gives us a baseline for what we can expect from released MEMS devices, and serves as a useful comparison point for evaluating losses and reflections in the circuits. Since the couplers and switch devices in these circuits are not in a pure cross or bar state, we observe the effects of multi-path interference and additional losses in the beam dumps. This is also anticipated with the MEMS devices when they are not actuated.

The measurement results show that none of the circuits on the packaged full demonstrator are working properly. Hence, to make sense of our observations the results are compared with the unprocessed chips. Similar to the full demonstrator switches, none of the switch circuits on both versions of mini demonstrators are operating, and most of the MEMS couplers have collapsed. For these mini demonstrator switches we have conducted measurements similar as for the full

[2] For this thesis, only the packaged chips for these specific circuits were available.

demonstrator, using the same experimental setup. Here are the steps:

1) Configuring the *Polatis* switch.

2) Programming the BeagleBone which sets the actuation voltages.

3) Running the python scripts to perform optical transmission and reflection measurements using *Luna* OVA.

We also have measured several etched and sealed chips; however, due to the collapse of the MEMS, the results were similar to those of the packaged chips. Therefore, they have not been included here to avoid redundant information.

6.3.1 Shunt Waveguides (on Fiber Arrays)

As discussed in Chapter 4 the optical I/O of the photonic chip is accommodated by two sets of 72-fiber arrays (fiber array A and B) near the edge of the chip. These fiber arrays include inputs and outputs of the demonstrator circuits such as switches and FP-PICs. They also include shunt waveguides which are used for the alignment of the 24-fiber array[3] to measure the optical response of the circuits. To address the grating couplers used in the 72-fiber arrays we refer to them as fn, where n is an integer number between 1 and 72; the schematic of the these fiber arrays and an image of them is shown in Fig. 6.8.

Figure 6.8 also shows the transmission of the different shunt waveguides on the fiber array A of the chips for both full and mini demonstrators. As seen, in the best situation transmission reaches $15\,\mathrm{dB}$. This confirms the proper attachment of the fiber arrays to the chips.

The reflection measurements of the fiber array A on the RUN2 full demonstrator chip are shown in Fig. 6.9 where the reflection peaks corresponding to the input and output grating couplers can be clearly seen. The shunt waveguides are simple rib waveguides connecting two grating couplers. They are not traversing a MEMS cavity and consequently have oxide clad waveguide.

As seen, in shunt waveguides with shorter lengths, the peak pairs are closer to each other. In fact, the distance between the input and output grating couplers (length of the shunt waveguides) can be determined by the distance between the peak pairs (indicated by the black triangles) or vice versa.

[3]Note that the 72-fiber arrays refer to the two sets of 72 grating couplers fabricated on the PICs, whereas the 24-fiber array consists of 24 closely packed fibers, as illustrated in Fig. 6.2.

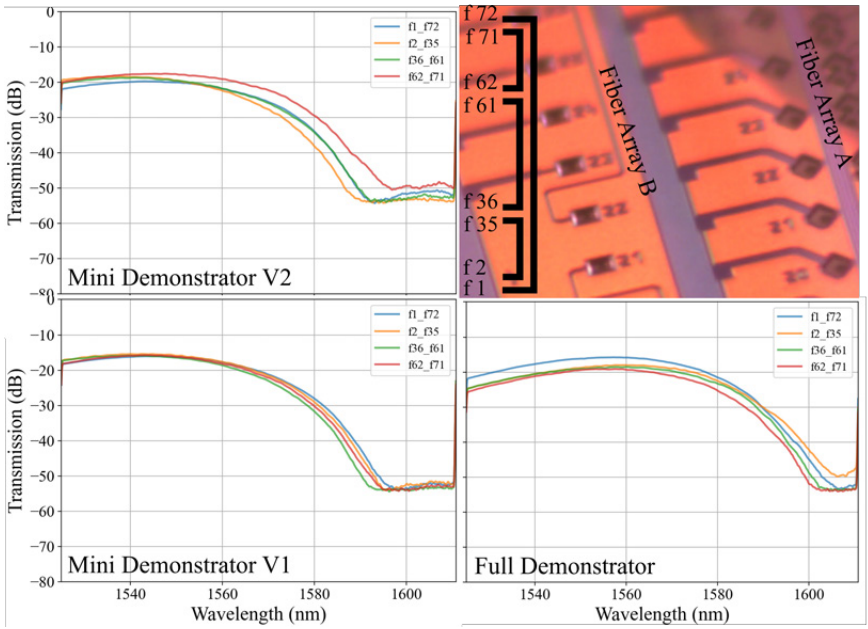


Figure 6.8: Transmission measurements of the Shunt waveguides of the fiber array A on the RUN2 full and mini demonstrators.

To check this, we measured the the distance between the input and output grating couplers from the GDS and LUNA reflection response. For instance, in the case of the $f1 - f72$ shunt waveguide, the measured lengths from the GDS and LUNA are $9993\,\mu m$ and $10\,048\,\mu m$, respectively (length difference of $55\,\mu m$). And, for the $f36 - f61$ shunt waveguide, the measured lengths from the GDS and LUNA are $3947\,\mu m$ and $3986\,\mu m$, respectively (length difference of $39\,\mu m$). In these calculations, we divided the LUNA data by 3.75, which represents the group index of the rib waveguide connecting the grating couplers. We use this approach to find the peaks corresponded to the circuit components as will be discussed in the next sections.

In Fig. 6.10 we have compared the reflection responses of the different shunt waveguides for various chips. Figure 6.10a shows the reflection response of the $f71$, from the $f62 - f71$ shunt waveguide, for the unprocessed chip and the full demonstrator in RUN2. As seen, they have similar responses as expected but the shunt waveguide on the unprocessed chip has a defect resulting in an extra peak in its response.

In Fig. 6.10b and Fig. 6.10c, we present the reflections from the input and output grating couplers ($f1$ and $f14$) of the shunt waveguide of the test nodes in

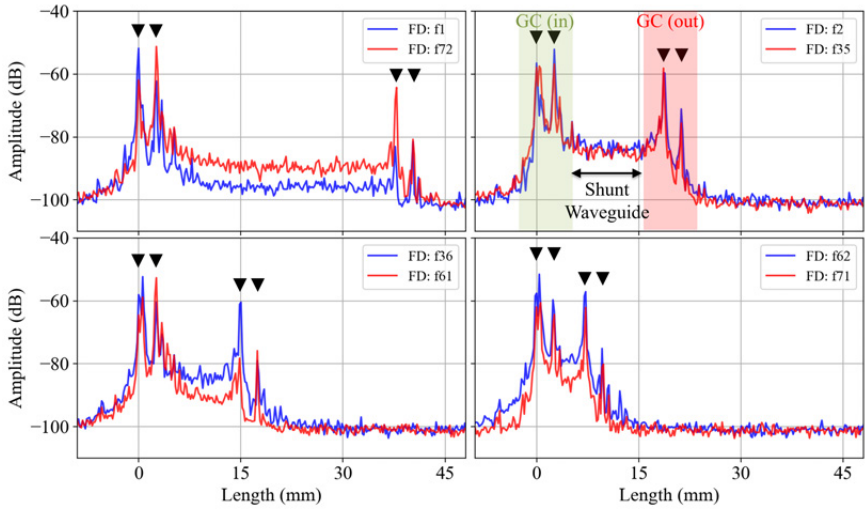


Figure 6.9: Reflection measurements of the Shunt waveguides of the fiber array A on the RUN2 full demonstrator chip. From the reflection impulse response, we clearly see the reflections of the grating couplers (GCs). Each GC has its own pair of the peaks indicated by the black triangles.

RUN3, comparing sealed and unprocessed chips. The figure highlights a defect near $f14$, which is evident in the reflection response of both $f1$ and $f14$.

To analyze the reflection response of longer shunt waveguides, we compared the $f1$ (from shunt waveguide $f1 - f72$) response on the both full and mini demonstrators. As observed, the response level in the waveguide region differs between FD and MD1, though they share a similar slope, indicating comparable waveguide loss. In contrast, FD and MD2 exhibit more similar responses. In Fig. 6.10f, all three packaged chips show a similar response at $f36$, within the $f36 - f61$ range, likely due to the shorter length of this shunt waveguide.

Another factor that could influence the reflection responses is the index-matching glue between the fiber facet and the chip. Since this glue prevents the fiber facet from reflecting light back, we expect packaged chips to have one fewer reflection peak compared to the hexapod measurements. Given the distance between the chip surface and the fiber facet (approximately $30\,\mu\mathrm{m}$), this peak should appear very close to the first peak in the reflection. However, it is difficult to identify on the response curve due to the $200\,\mu\mathrm{m}$ resolution of the LUNA.

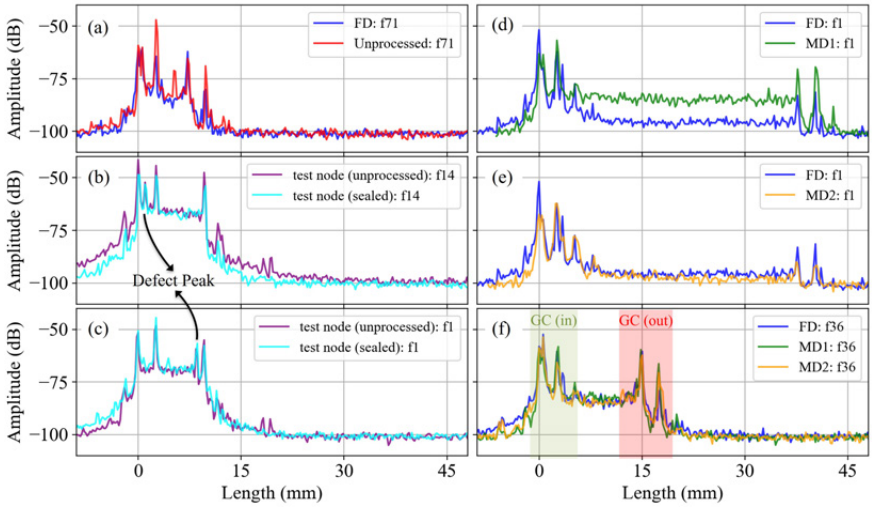


Figure 6.10: Reflection measurements of the Shunt waveguides of the fiber array A on different chips. a) $f71$ for the full demonstrator and an unprocessed chip for RUN2. b,c) $f14$ from test nodes on the unprocessed and sealed chips for RUN3. d,e) Comparison of the $f1$ for the full and mini demonstrators. f) Comparison of the $f36$ for the full and mini demonstrators.

6.3.2 Crossbar 4×4 switch

Figure 6.11 shows the layout, schematics, and the port maps of the Crossbar 4×4 switch circuit designed for RUN2. This circuit contains 16 MEMS couplers, 4 inputs, and 4 outputs where the inputs are on the North side of circuit and outputs are on the East side.

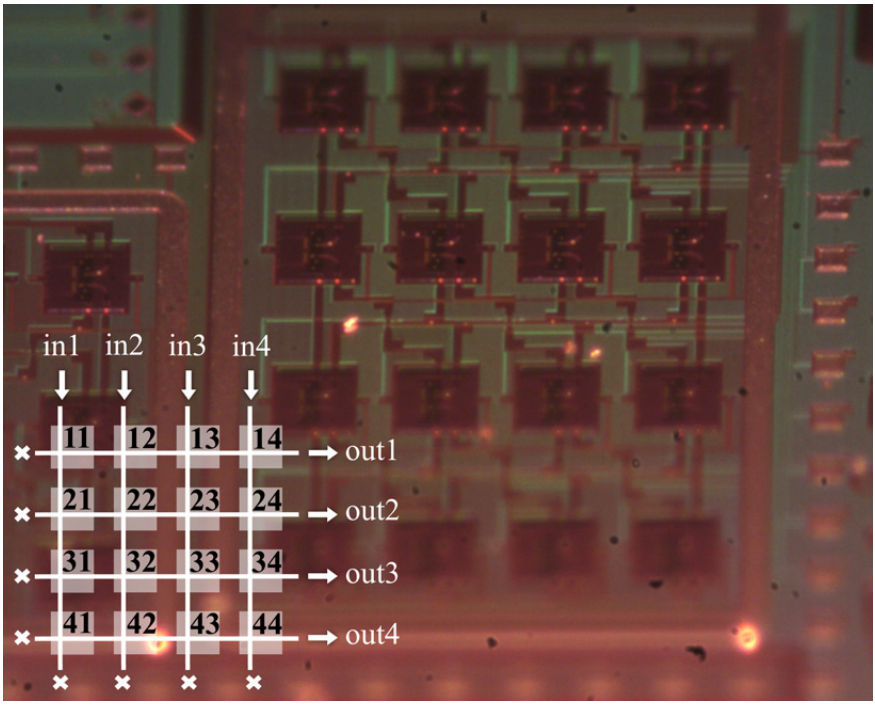


Figure 6.11: Crossbar 4×4 switch circuit fabricated on MORPHIC RUN2.

6.3.2.1 Unprocessed Chip

The measured transmissions of the circuit for all 16 ports combinations of the Crossbar 4×4 have been summarized in Fig. 6.12a-d. For better understanding we have shown the light path of each port combination on the circuit schematic in Fig. 6.12e, where the blue, orange, green, and red colors correspond to the paths going to the output 1, 2, 3, and 4 accordingly. As clearly seen, there is no interference pattern in the transmissions which indicates there is no propagation along multiple paths, there is always only one way to connect a selected input to a desired output. The highest measured transmission belongs to the $in4 - out1$ pair, as there is only one tunable coupler in the path (Fig. 6.12d).

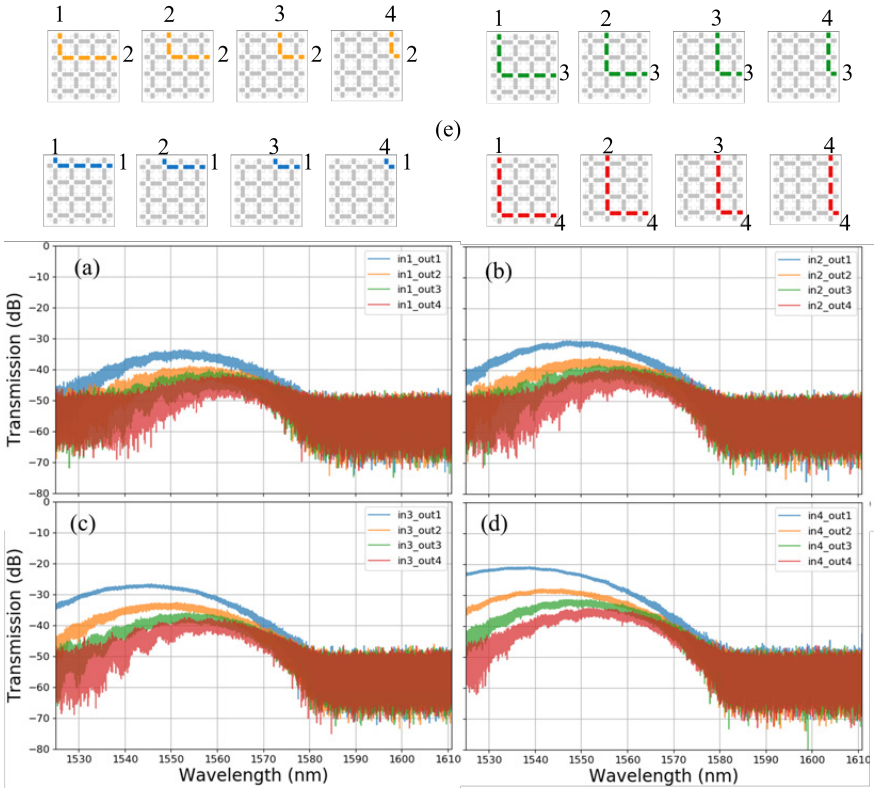


Figure 6.12: Transmission measurements of the Crossbar 4×4 switch circuit on an unprocessed RUN2 chip for various input and output combinations. (a-d) all 16 combinations of input and output ports. (e) Possible light paths when going from the selected input to the outputs.

Figure 6.13 shows the reflection measurements in which the reflections of the grating couplers (GC), the four switches, and the output grating coupler are clearly identified. For the grating couplers, multiple peaks are observed while it is expected to see two peaks. The distance between these peaks is quite uniform. So the light travels again the same length. This could be due to multiple reflections from the substrate and the top surface of the chip (or because of a Fabry-Perot cavity between the top and bottom of the silicon substrate). Possibly there are some reflections from the fiber facets or the interface of the grating and the waveguide. Since the directional couplers used in the switch are designed in the partial state (neither Bar nor Cross state), the incoming light from each input may split over different paths and as a result passes through additional switch devices. This can result in multiple peaks with different distances (indicated by the red box in Fig. 6.13) in the

measured reflection response and induces more loss.

For a better understanding, the schematic of all the possible paths toward outputs for the selected input ports are shown in Fig. 6.13. As an example, $in1$ has to traverse more switch devices to reach $out4$ compared to the port $in4$, hence it has more peaks in its reflection plot. These additional MEMS devices also induce more loss, and as a result we can see that the reflection peak of the output grating coupler becomes weaker for the longer paths.

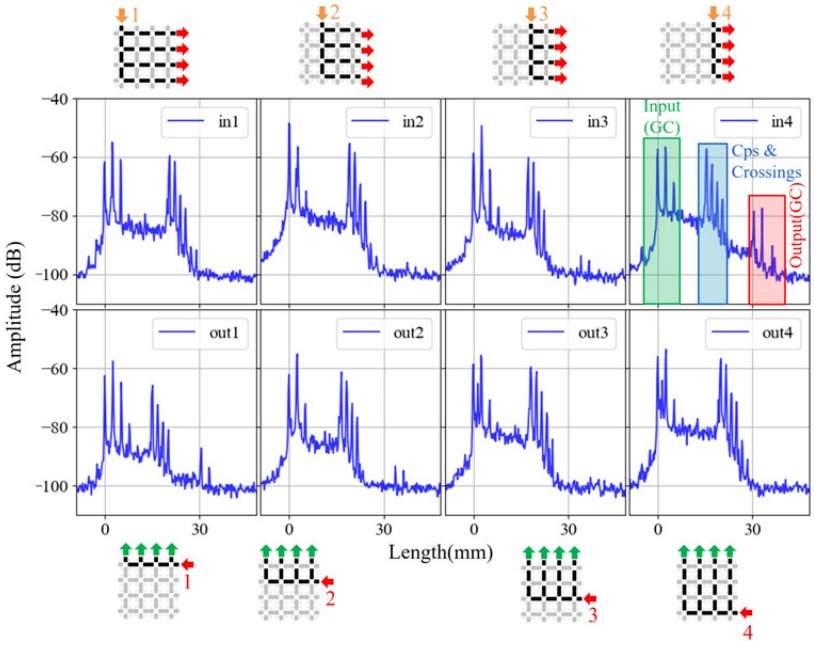


Figure 6.13: Reflection measurements on the Crossbar 4×4 switch circuit on an unprocessed chip. From the reflection impulse response, we clearly see the reflections of the grating couplers, the four switches and the output grating couplers. Schematics of all possible light paths when going from the selected port to the others.

As explained earlier, the reflection includes all losses through the circuit twice, and partial transmission through a coupler will, in this case, contribute to loss. These switch devices are constructed in such a way that they only have low loss in pure bar state and pure cross state: As they make use of two directional couplers in series, a partial state will send light into a 'dump' waveguide in the MEMS device, or towards the dump ports (which have an absorbing termination) in the South of the circuit (for the device description see Chapter 3). Hence, the losses of this pure passive, unactuated measurement are higher than if the switch devices would be in their optimal state. Additionally, inverse Fourier analysis of the Luna OVA for the

transmission between the $in1 - out1$ and $in1 - out4$ pairs show that there is no interference, and only one peak for each path is seen (Fig. 6.14).

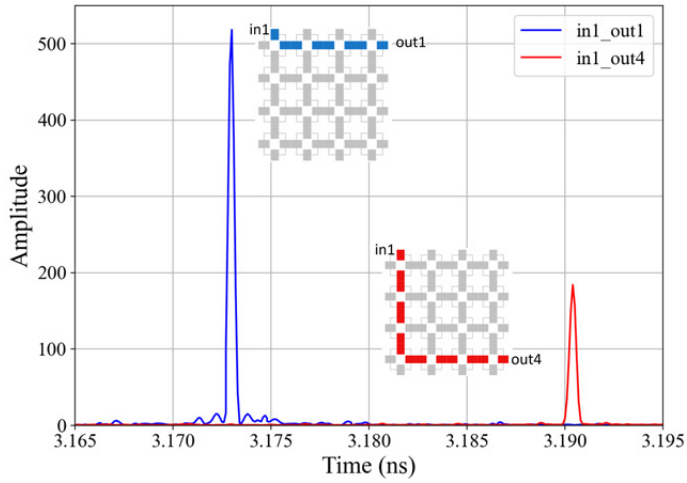


Figure 6.14: Identification of the transmitted powers in possible paths using the inverse Fourier analysis of the Luna OVA for Crossbar 4×4 switch circuit on an unprocessed chip, for $in1 - out1$ and $in1 - out4$ pairs. Schematics show all the possible light paths for the selected 4×4 input and output.

6.3.2.2 Packaged Chip: Full Demonstrator

The measured transmission and reflection results of the Crossbar 4×4 switch circuit on the full demonstrator chip have have been summarized in Fig. 6.15 and Fig. 6.16, respectively. As seen, the best transmission response belongs to the $in4 - out1$ pair which has only one coupler in its light path (coupler 14 in Fig. 6.11). Compared to the transmission of the Shunt waveguide, this path has an insertion loss of only $-23\,\mathrm{dB}$, even without tuning. This shows that this coupler is largely in cross state. Indeed, reflection measurement in Fig. 6.16 clearly shows the reflection peaks of the output grating coupler when injecting light in ports $in4$ and $out1$. It can also be seen that some light from $in4$ goes down to the other switches, as (increasingly weak) transmission is seen in ports $out2$, $out3$ and $out4$.

Comparing transmission response of this circuit with the one on the unprocessed chip, shows an overall lower transmission of the circuits, and several paths without any transmission at all. This can be attributed to the collapsed/bridged MEMS couplers. Moreover, there are oscillations on some spectra that were not present in the unprocessed chips. These can have two causes: multi-path interference, or multiple backreflections that create a weak Fabry-Pérot cavity. As the circuit

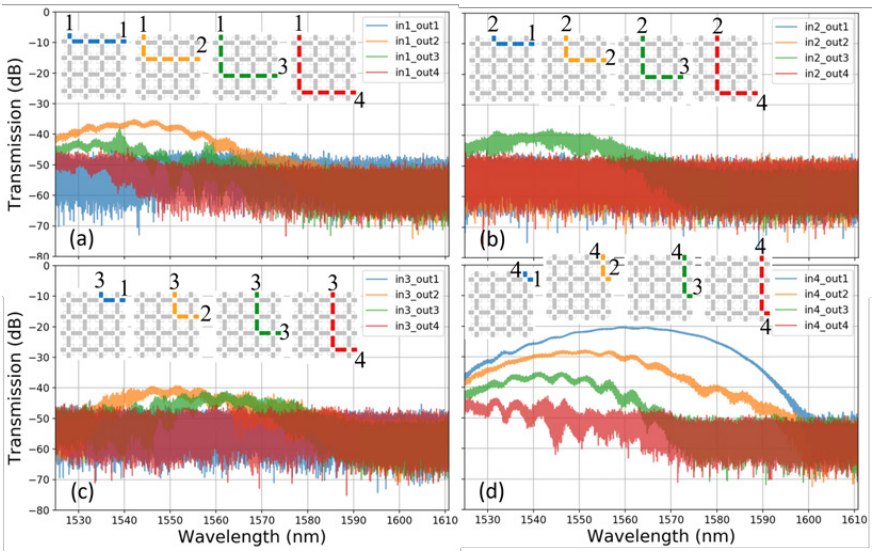


Figure 6.15: Transmission measurements of the Crossbar 4×4 switch circuit on the full demonstrator for various input and output combinations. (a-d) All 16 combinations of input and output ports. Schematics show possible light paths when going from the selected input on the top side to the outputs on the right side.

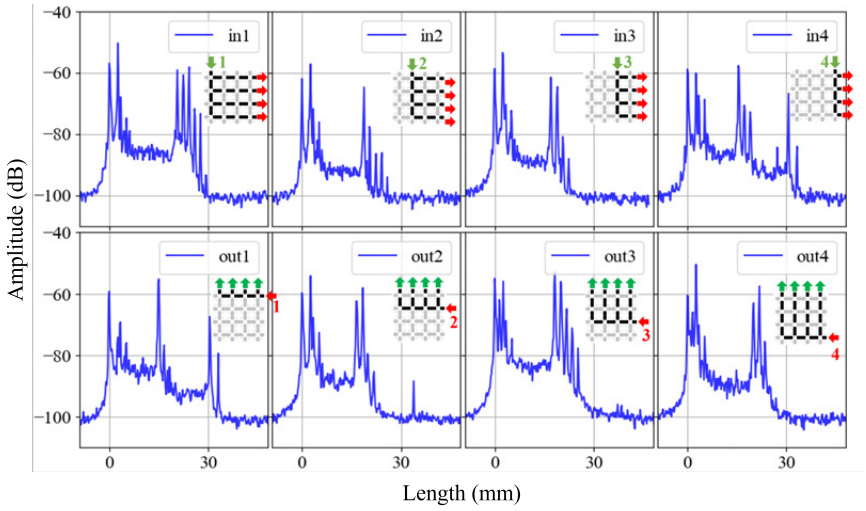


Figure 6.16: Reflection measurements on the Crossbar 4×4 switch circuit on the full demonstrator. From the reflection impulse response, we can see that only in4 and out1 have strong reflection peaks for the output grating coupler.

itself does not really support multiple connection paths, it can be expected that the backreflections are stronger and therefore give rise to such weak resonances. In this circuit, it looks like not too many of the MEMS devices have collapsed, as there is measurable transmission between many input-output pairs. This does not yet mean that all devices are operational. These switches were designed with a 150 nm gap, and therefore are prone to bridging, i.e. small silicon or alumina fragments in the gap of the directional coupler that mechanically connect the two arms of the coupler together. To see if the switches respond, we will have to apply actuation (Sec. 6.4.1).

6.3.2.3 Packaged Chip: Mini Demonstrator

Figure 6.17 shows the transmission measurements of the Crossbar 4×4. As seen, the best response is for the $in4-out1$ pair which its transmission can only reach $-40\,\mathrm{dB}$. Also, a small portion of the light can reach the outputs when the light is injected in port $in1$. This indicates that most switches have collapsed (which puts them in bar state), except for the switches in the Westmost column, which still couple some light (but not much) to the cross state. Also, the North-East coupler (41) seems to couple some light to the cross state.

Reflection measurements also don't show any clear sign of light reaching the output grating couplers (Fig. 6.18). The existence of multiple peaks in the reflection response indicates that light can penetrate into the circuits for some of the ports. However, lack of overall transmissions shows that MEMS have high losses which can be attributed to the out-of-plane movements (such as a collapse) or bridging.

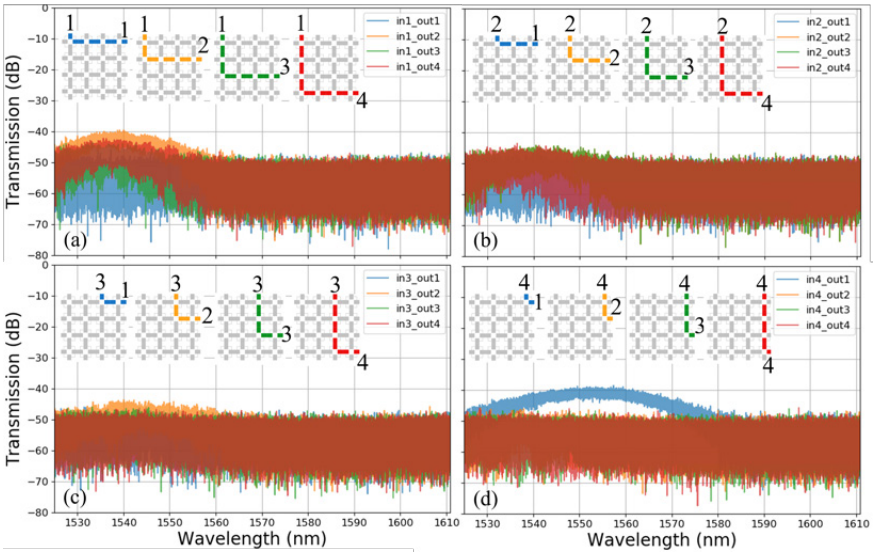


Figure 6.17: Transmission measurements of the Crossbar 4×4 switch circuit on the mini demonstrator (version 2) chip for various input and output combinations. (a-d) all 16 combinations of input and output ports. We see than most devices have collapsed and therefore do not transmit any light.

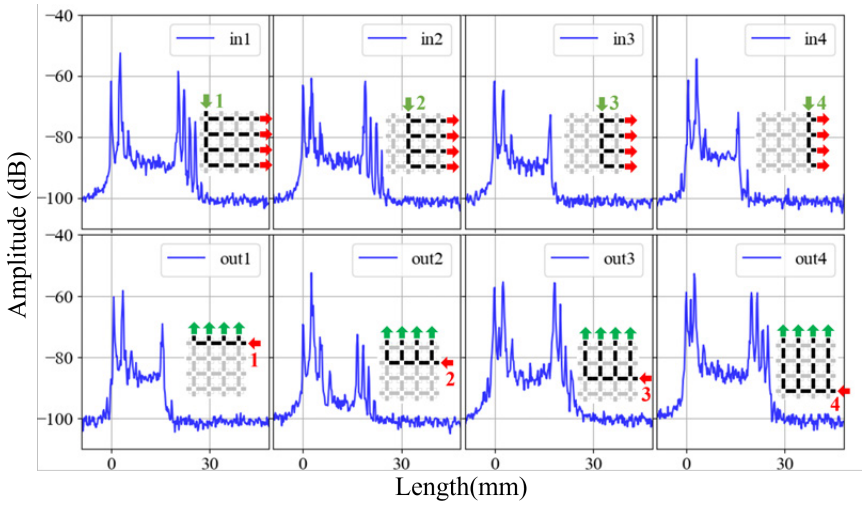


Figure 6.18: Reflection measurements on the Crossbar 4×4 switch circuit on the mini demonstrator (version 2) chip. Schematics of all possible light paths when going from the selected port to the others.

6.3.3 PI-loss 4×4 switch

The next circuit for our investigation is a PI-loss 4×4 switch, for which the fabricated circuit and schematic have been shown in Fig. 6.19. It contains 4 inputs (West side) and 4 outputs (East side), and the number of the used couplers is 16, like with the crossbar switch. This circuit also has 4 dump ports, but unlike the Crossbar 4×4, it allows multi-path interference, as will be discussed further.

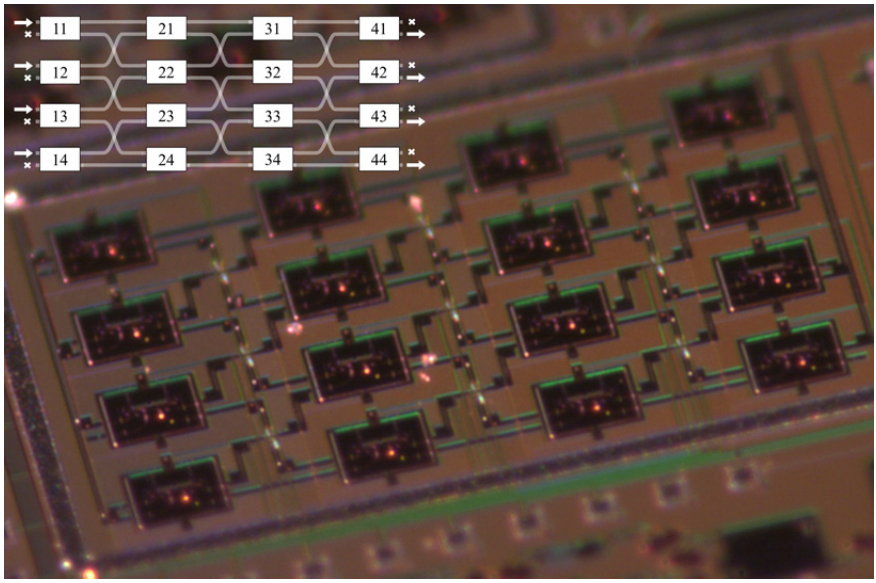


Figure 6.19: PI-loss 4×4 switch circuit for RUN2. (a) Fabricated Circuit (b) Schematic.

6.3.3.1 Unprocessed Chip

The measured transmissions of the circuit for all 16 port combinations have been summarized in Fig. 6.20a-d. For better understanding, the light path of each port combination on the circuit schematic is shown in Fig. 6.20e, where the blue, orange, green, and red colors correspond to the paths going to the output 1, 2, 3, and 4 accordingly.

Compared to the Crossbar 4×4 switch circuits, more transmissions are reaching the $-20\,\mathrm{dB}$ level, indicating better loss performance of this circuit. There is, at this point, no state of 'path independent loss' as the name of the switch suggests, because the couplers are not in perfect bar state or cross state, and therefore some light is sent to the 'dump' ports in the East. As seen, the highest measured transmission belongs to the pairs $in1-out1$, $in2-out3$, $in3-out4$, $in4-out3$, and $in4-out4$.

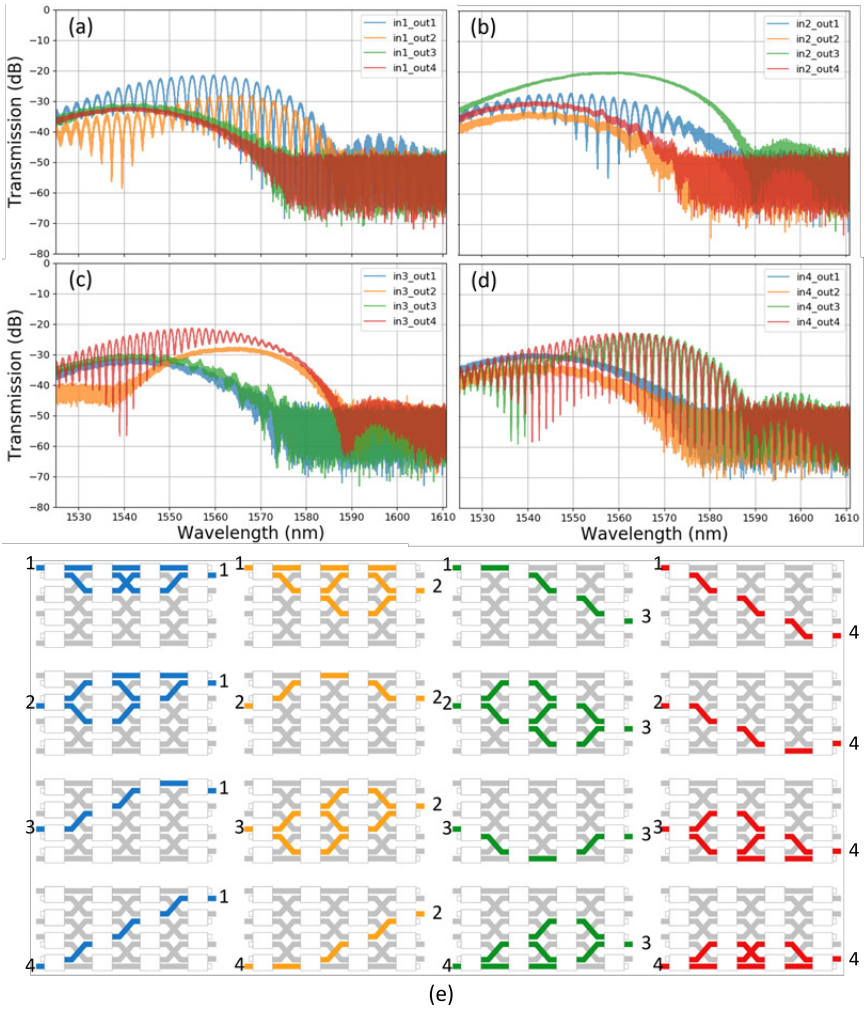


Figure 6.20: Transmission measurements of the PI-loss 4×4 switch circuit on an unprocessed RUN2 chip for various input and output combinations. (a-d) all 16 combinations of input and output ports. (e) Possible light paths when going from the selected input to the outputs (inputs are on the left side of the circuit and outputs are on the right side).

Also, unlike the previous circuit, light interference pattern in the transmissions indicates several ways with a different path length between each selected input and its corresponding desired output (Fig. 6.20e). These are all the combinations that include a straight path along the North or South border of the switch. Examples such as $in2 - out3$ or $in3 - out2$ also have multiple paths connecting the ports, but

there all paths are designed to be the same length. Even if there are small optical path length differences due to fabrication variations, these will result in a broadband response.

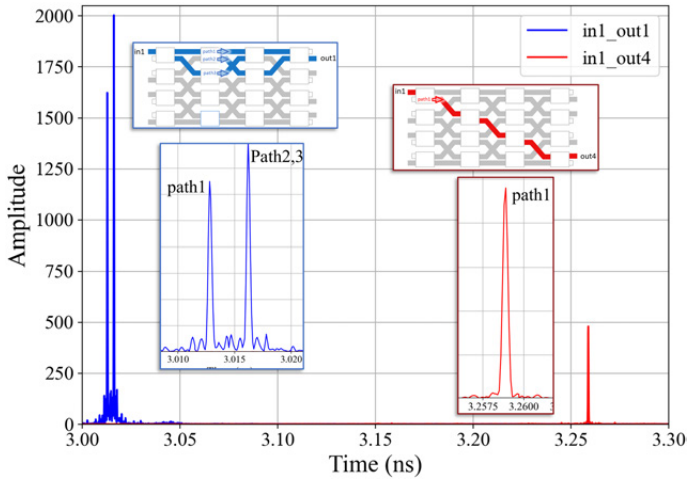


Figure 6.21: Identification of the transmitted powers in possible paths using the inverse Fourier analysis of the Luna OVA for PI-loss 4×4 switch circuit on an unprocessed chip, for $in1 - out1$ and $in1 - out4$. Schematics show all the possible light paths for the selected input and output.

Looking at the spectra with fringes in the time domain (inverse Fourier transform) helps us to clearly identify two pulses arriving with a delay. For instance, see Fig. 6.21 for comparison of $in1 - out1$ and $in1 - out4$. As seen, the $in1 - out1$ pair can be constructed by using three paths, of which two have the same length; on the other side, the pair $in1 - out4$ can have only one connection path. This is an important phenomenon to keep in mind when analyzing released MEMS devices: a lack of interference when it is expected can be the sign of a collapsed MEMS along one of the possible paths.

Figure 6.22 shows the reflection measurement results of the PI-loss 4×4 switch circuit. Reflections of the grating couplers, the switches, and the output grating coupler are obvious. Note that, because the switches are in a partial coupling state, each port reaches many more than four switches at different distances, so the reflection response shows more peaks compared to the Crossbar 4×4. As seen in Fig. 6.13, red and blue colors are used for the couplers in the schematics shown in this figure. This gives us better visualization to identify couplers which can be involved in the interference mechanism. In some cases ($in1$ and $out2$), output grating couplers peaks cannot be seen which indicates higher value of the loss in their related paths. One of the possible scenarios could be destructive interference

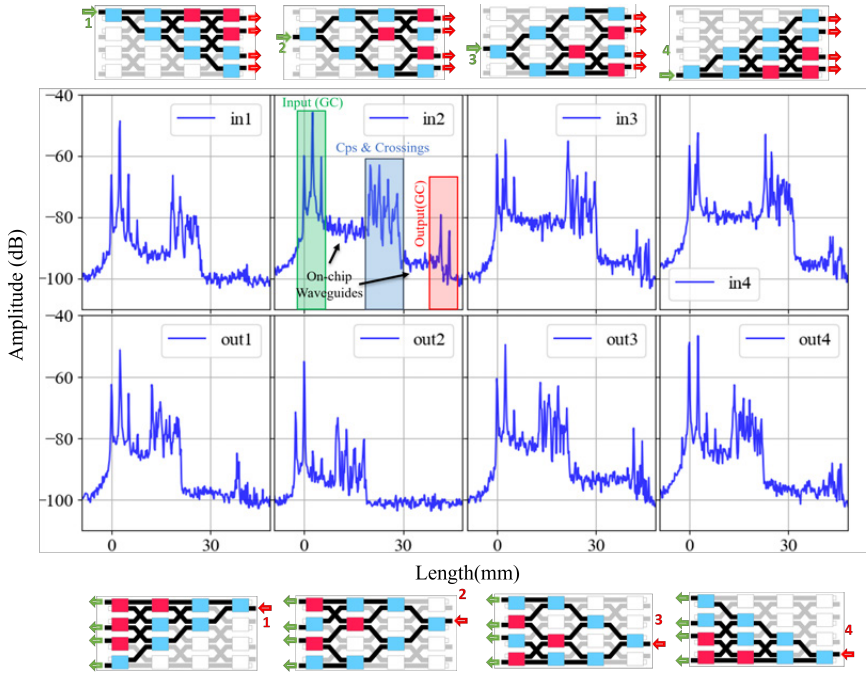


Figure 6.22: Reflection measurements on the PI-loss 4×4 switch circuit on an unprocessed chip. From the reflection impulse response, we clearly see the reflections of the grating couplers, switches, and the output grating couplers. For each plot, we show the schematics of all possible light paths when going from a selected port to the others. Red couplers are involved in light interference, but the blue ones only act as a splitter.

of the reflected light from the couplers and crossings.

6.3.3.2 Packaged Chip: Full Demonstrator

For the PI-loss 4×4 switch circuit, the situation seems to be worse: only $in4 - out3$ and $in4 - out4$ pairs show transmission (Fig. 6.23). Note that, as the inset schematics indicate, there are multiple paths between these ports. Yet, interference fringes are not observed, indicating that only a single path survives between these port combinations.

It is not surprising that the surviving paths are in the South of the circuit. The tunable coupler elements used in this circuit are asymmetric: as can be seen in Fig. 6.24, the South arm is a fixed rigid waveguide, while the actuated North arm consists of flexible S-bends and a suspended comb drive. It is clear that this North arm is much more prone to collapse. In that case, the South ports of the coupler

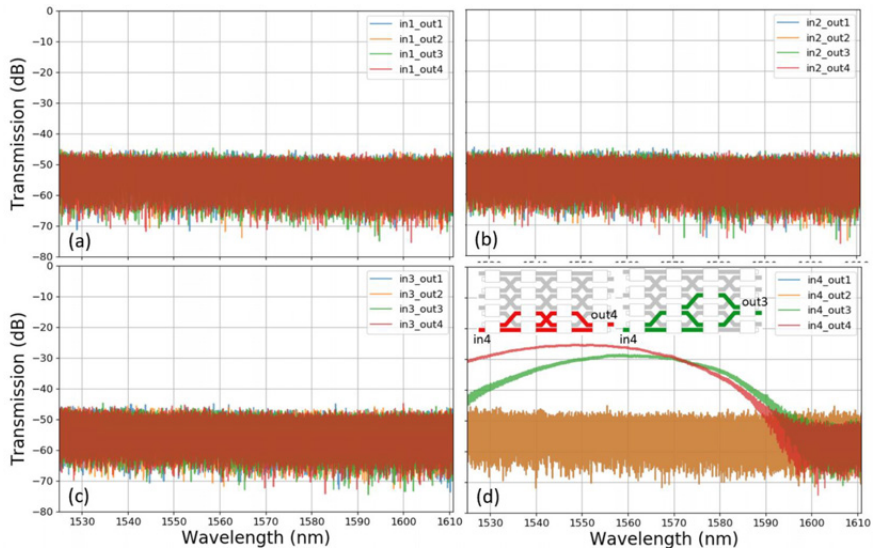


Figure 6.23: Transmission measurements of the PI-loss 4×4 switch circuit on the full demonstrator chip for all input and output combinations.

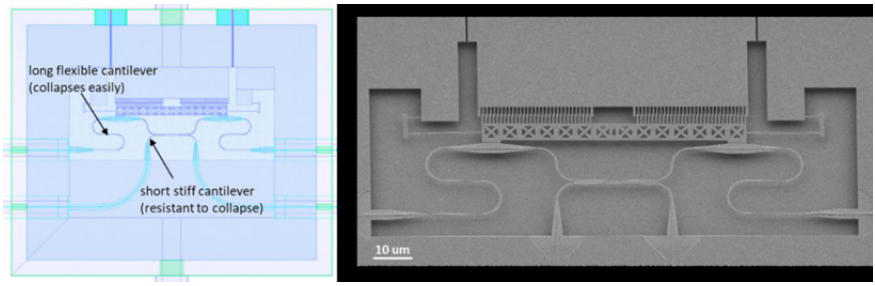


Figure 6.24: Switch devices used in PI-loss and Benes switch circuits [10]. The switch is asymmetric, with a flexible arm (North) and a rigid arm (South). The flexible arm is more prone to collapse.

are in 'bar' state while the North ports are blocked. What might prevent such a collapse is of course bridging in the gap between the two waveguides, in which case the coupler is in a partial coupling state and the bridging might also induce some scattering and backreflection. Hence, from the measurements in the PI-loss circuit, the most likely deduction is that the third coupler in the South row (coupler 43 in Fig. 6.19) is in a partial state, while the others have collapsed.

Reflection analysis also confirms this (Fig. 6.25): only three ports show reflection peaks of output grating couplers: $in4$, $out3$, and $out4$. We also see that injection in $in1$, $in3$ and $out2$ shows only a single reflection peak within the circuit.

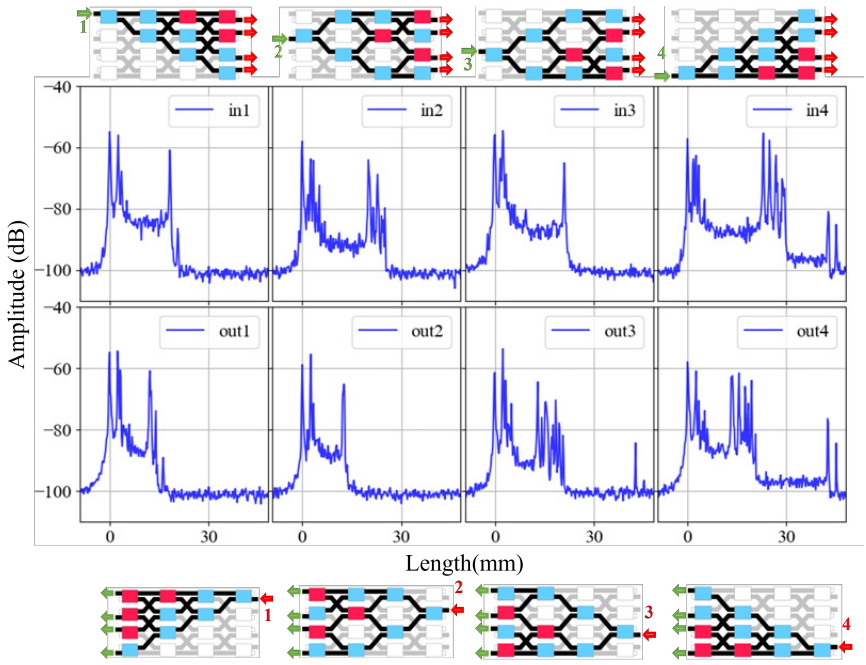


Figure 6.25: Reflection measurements on the PI-loss 4×4 switch circuit on the full demonstrator chip. From the reflection impulse response, we can see that only $in4$, $out3$, and $out4$ have strong reflection peaks for the output grating coupler. Schematics of all possible light paths when going from the selected port to the others.

And for the remaining ports, we get some additional peaks, but the count never goes up to 4, indicating that the light cannot reach the other side of the circuit.

6.3.3.3 Packaged Chip: Mini Demonstrator

The PI-loss switch circuit on the mini demonstrator also suffers from nonfunctional MEMS. No transmission is observed for all port combinations (Fig. 6.26), and the reflection measurements (Fig. 6.27) only show single peaks in the couplers region, which indicates that light has not even able to reach the second couplers. Even for port $in4$, which is connected to a path that contains only rigid waveguides, only a single pronounced reflection peak is visible.

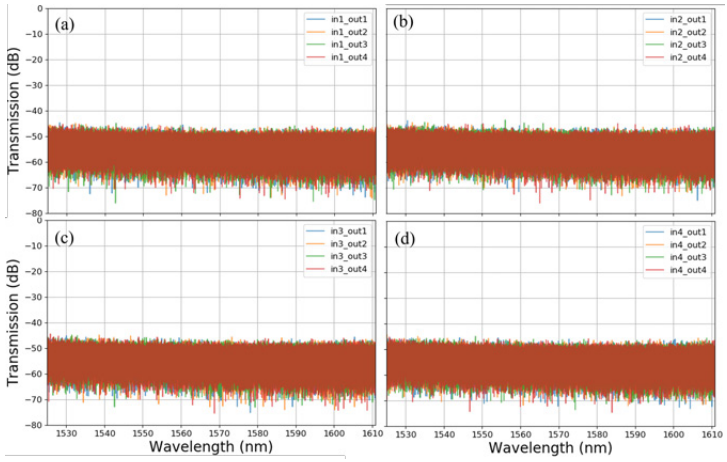


Figure 6.26: Transmission measurements of the PI-loss 4×4 switch circuit on the mini demonstrator (version 2) chip for various input and output combinations. (a-d) all 16 combinations of input and output ports.

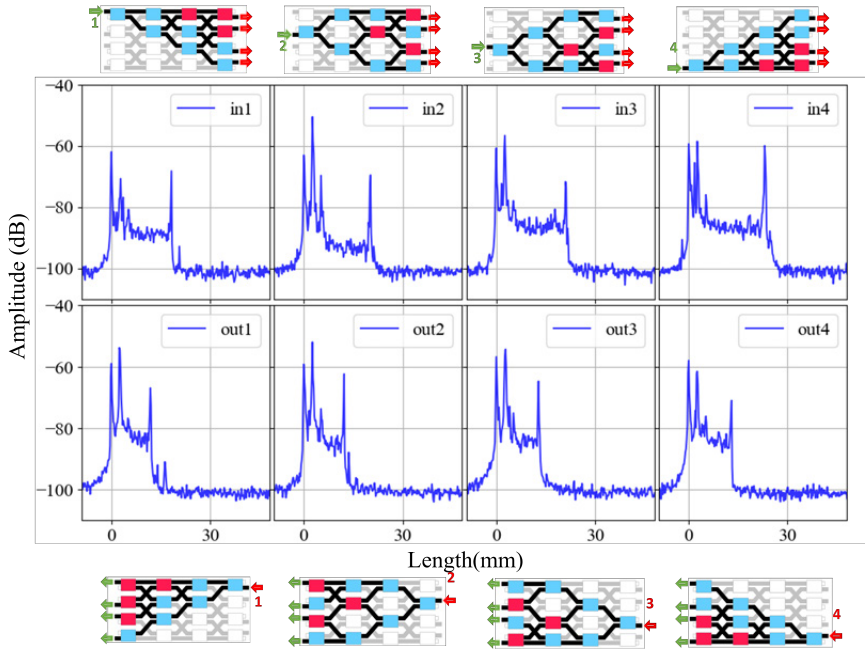


Figure 6.27: Reflection measurements on the PI-loss 4×4 switch circuit on the mini demonstrator (version 2) chip. And, schematics of all possible light paths when going from the selected port to the others.

6.3.4 Benes 4×4 switch

The simplest switch designed in RUN2 is the Benes 4×4 switch circuit (Fig. 6.28). It has 6 tunable couplers (the same type used in the PI-loss switch), 4 inputs (left side) and 4 outputs (right side). In contrast to the Crossbar 4×4 and the PI-loss 4×4, this circuit does not have dead-end beam dumps. In other words, all light injected in the circuit should reach the end, except for propagation losses, reflections, and collapsed MEMS devices along the way.

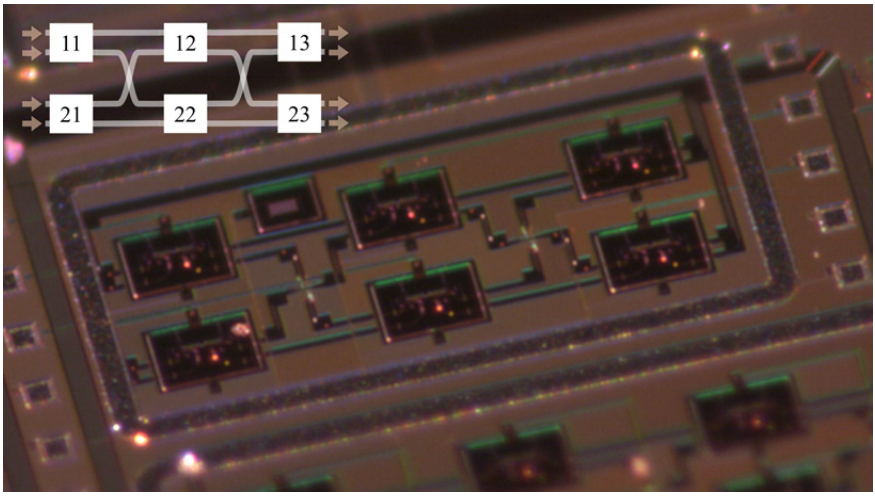


Figure 6.28: Benes 4×4 switch circuit on MORPHIC RUN2 chip.

6.3.4.1 Unprocessed Chip

Transmission measurement results of this circuit have been summarized in Fig. 6.29. As expected, it has better loss performance because of the lack of beam dumps and the lower number of the couplers the light has to pass through. We can also see two different interference patterns (with a different FSR). This can also be observed in the schematics where blue, orange, green, and red paths correspond to the paths going toward output ports 1, 2, 3, and 4.

For better comparison of the interference patterns we have plotted transmissions of all inputs to the $out1$ in Fig. 6.30. As demonstrated by the schematics, there are the two possible interfering paths for the light injected to each port to reach $out1$, forming an MZI for all cases. Hence, the Free Spectral Range (FSR)[4] and

[4] $FSR = \lambda^2/(n_g \times \Delta L)$, where λ is the operating wavelength, n_g is the group index of the waveguide, and ΔL is the physical path length difference between the two paths.

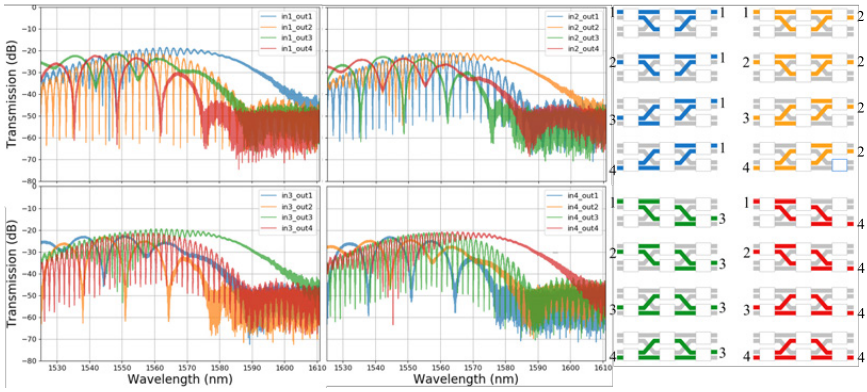


Figure 6.29: Transmission measurements of the Benes 4×4 switch circuit on an unprocessed RUN2 chip for various input and output combinations. (a-d) all 16 combinations of input and output ports. (e) Possible light paths when going from the selected input to the outputs (inputs are on the left side of the circuit and outputs are on the right side, see Fig. 6.28

Extinction Ratio (ER) of the responses are included in the figure. FSR is the wavelength spacing between successive transmission dips. It is determined by the difference in optical path lengths between the two arms of the interferometer. And, ER is the ratio of the maximum transmitted power to the minimum transmitted power. High ER occurs when the splitting ratio between the two paths is close to 50:50, and phase differences lead to nearly perfect constructive and destructive

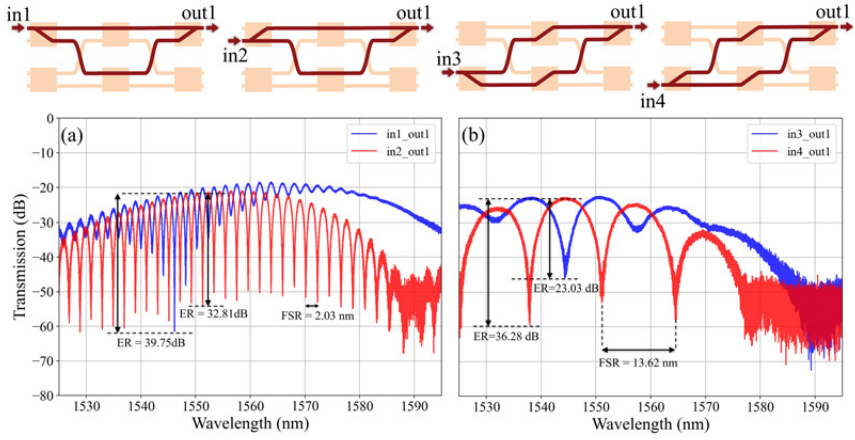


Figure 6.30: Transmission measurements of the Benes 4×4 switch circuit on an unprocessed RUN2 chip for all inputs to the $out1$: a) $in1 - out1$ and $in2 - out1$, b) $in3 - out1$ and $in4 - out1$. The schematic of the all light paths for each input and $out1$ combination is also shown.

interference. Low ER happens when the splitting is imperfect, leading to residual light even during destructive interference.

As expected, $in1 - out1$ and $in2 - out1$ transmissions have higher FSR compared to the $in3 - out1$ and $in4 - out1$ because of the longer path differences between their possible two paths (see the schematics). Additionally, high extinction ratio (ER) of the responses suggests that couplers should have the split ratio of 50:50. For example, the ER of the $in1_o ut1$ reaches to $40\,\mathrm{dB}$

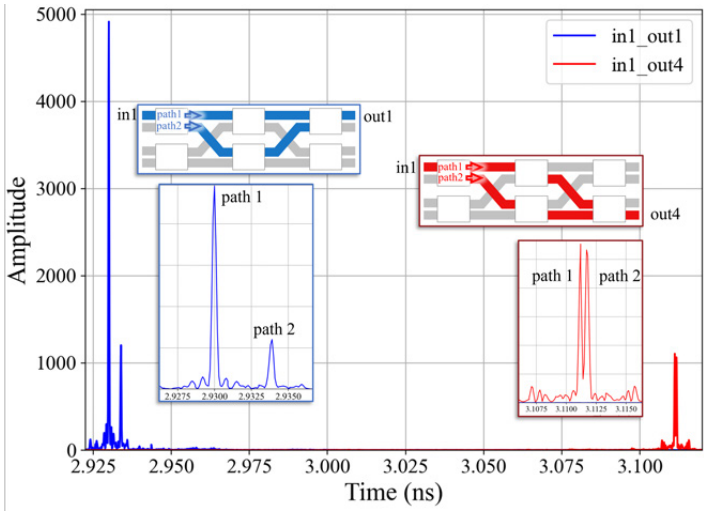


Figure 6.31: Identification of the transmitted powers in possible paths using the inverse Fourier analysis of the Luna OVA for Benes 4×4 switch circuit on an unprocessed chip for $in1 - out1$ and $in1 - out4$. Schematics show all the possible light paths for the selected input and output.

In Fig. 6.31, we have compared two ports pairs of $in1 - out1$ and $in1 - out4$. The first one has a larger path difference while the other has a smaller one. This can be seen from the distance between the reflection peaks obtained from the inverse Fourier transforms. In fact, the path length difference in the second case is only because the switch elements have an asymmetric layout. If they would have been designed symmetric, the two paths would be identical by design, and there would only be a small phase difference, and thus a broadband interference, because of small fabrication variations. Next, a grid for the measured reflections of the Benes 4×4 switch circuit is plotted in Fig. 6.32. Compared to the previous switches the reflection peaks of all the output grating couplers can be seen and they are higher in number (each input goes to multiple outputs). Also, the three peaks related to the couplers are clearly seen, other small peaks can be attributed to the waveguide crossings.

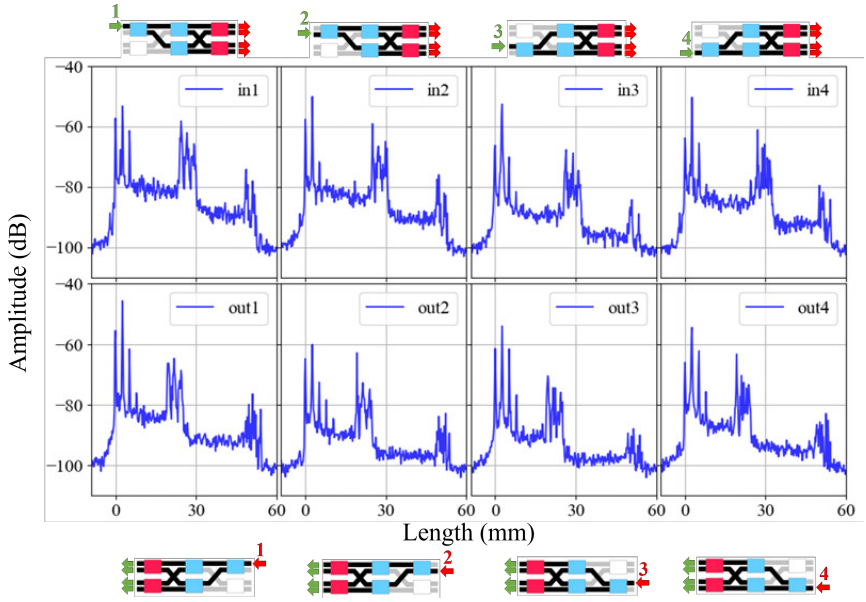


Figure 6.32: Reflection measurements on the Benes 4×4 switch circuit on an unprocessed chip. From the reflection impulse response, we clearly see the reflections of the grating couplers, the three switches and the output grating couplers. The schematics show all possible light paths when going from the selected port to the others. Red couplers are involved in light interference, but the blue ones only act as a splitter.

6.3.4.2 Packaged Chip: Full Demonstrator

Similar to the PI-loss 4×4, the Benes 4×4 in the full demonstrator has a poor transmission (Fig. 6.33 and 6.34). It only shows two successful transmissions for the $in3 - out4$, and $in4 - out4$ pairs.

While the reflection measurement results in Fig. 6.34 show that only $in3$, $in4$ and $out4$ can transmit light, we can observe the three peaks of the switches and the output grating coupler. For the other ports, light can penetrate only one or two couplers deep into the circuit. Note that this Benes switch circuit uses the same tunable coupler devices as the PI-loss. So here, too, it is not surprising that the South paths are the ones that are most likely to survive. The most likely explanation for the observed transmission and reflection is that the South-West coupler (coupler 21 in Fig. 6.28) is in a partial state (possible bridged) and that the North paths of the other couplers have all collapsed.

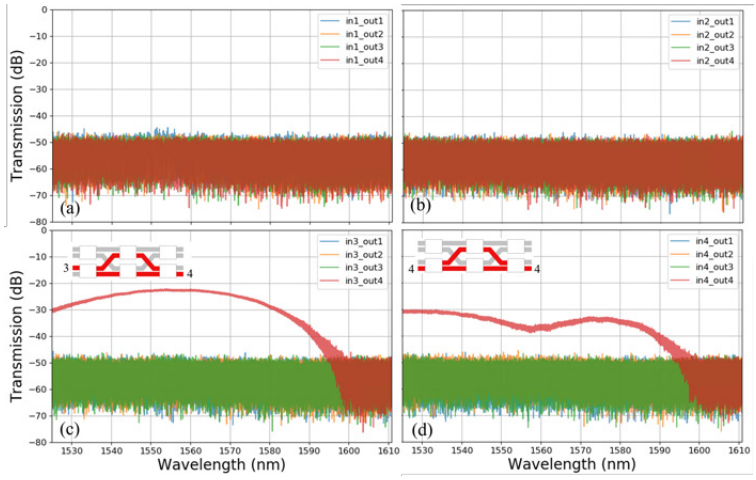


Figure 6.33: Transmission measurements of the Benes 4×4 switch circuit on the full demonstrator chip for various input and output combinations.

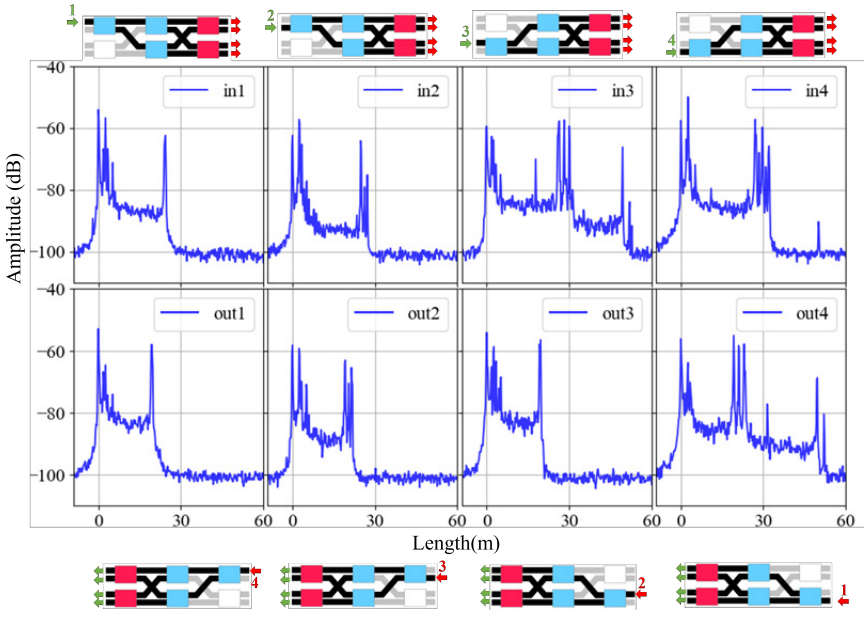


Figure 6.34: Reflection measurements on the Benes 4×4 switch circuit on the full demonstrator chip. From the reflection impulse response, it is seen that only $in3$, $in4$, and $out4$ have strong reflection peaks for the output grating coupler. Schematics of all possible light paths when going from the selected port to the others.

6.3.4.3 Packaged Chip: Mini Demonstrator

As previously mentioned, the Benes 4×4 switch circuit uses the same coupler devices as the PI-loss 4×4 circuit. Hence, it faces the same issue: no observable transmission response, which is seen for the mini demonstrator chip as well. As shown in Fig. 6.35 we only can see a weak response from the $in4 - out4$ pair. This is, of course, the only path with only rigid couplers, and yet such high loss is seen.

Similar to previous cases we have also shown the reflection response of this chip (Fig. 6.36). Comparing the reflection response of the odd and even ports shows that most of the odd ports have only a single peak in the circuit while the even ports demonstrate multiple peaks. In fact, the odd ports are all connected to a flexible and non-rigid arm of the couplers which can easily collapse as seen. On the other side, even ports are more rigid and the light can at least travers through the first port.

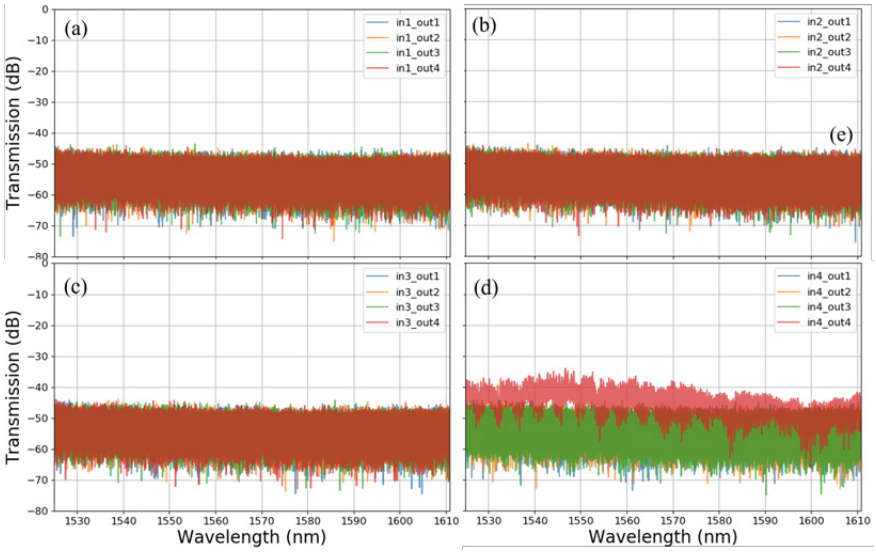


Figure 6.35: Transmission measurements of the Benes 4×4 switch circuit on the mini demonstrator (version 2) chip for various input and output combinations. (a-d) all 16 combinations of input and output ports.

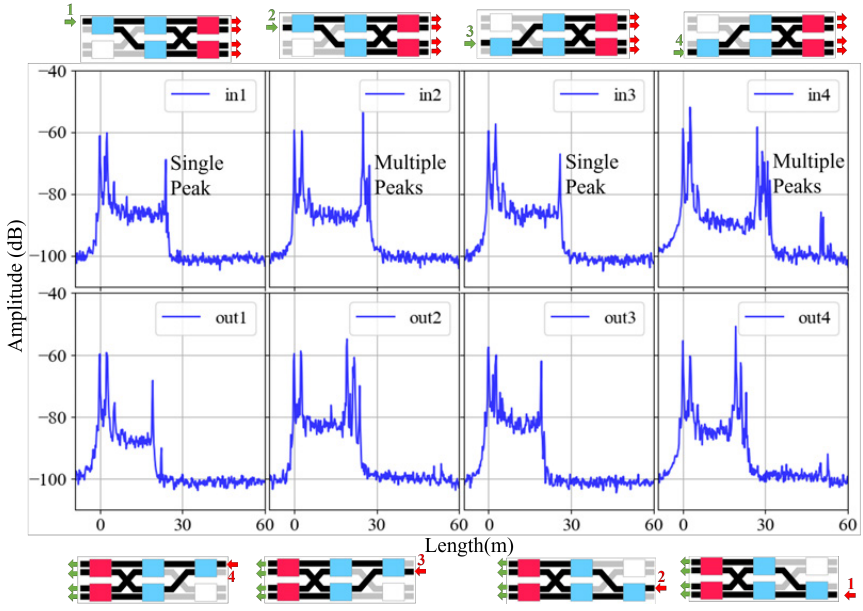


Figure 6.36: Reflection measurements on the Benes 4×4 switch circuit on the mini demonstrator (version 2) chip, and schematics of all possible light paths when going from the selected port to the others.

6.3.5 Benes 16×16 switch

The Benes 16×16 is the largest switch that has been designed in RUN2 (Fig. 6.37). It has 16 inputs (West side) and 16 outputs (East side) and has 56 couplers. Therefore, the most chaotic response is expected when all the couplers are in partial state. For this circuit measurement we used a 24-fiber array. As a result, we could only measure combination of input ports of $1-16$ and output ports of $9-16$. Like the Benes 4×4 switch, this circuit has no beam dumps. All light is transmitted to the output ports, or otherwise reflected or lost during propagation.

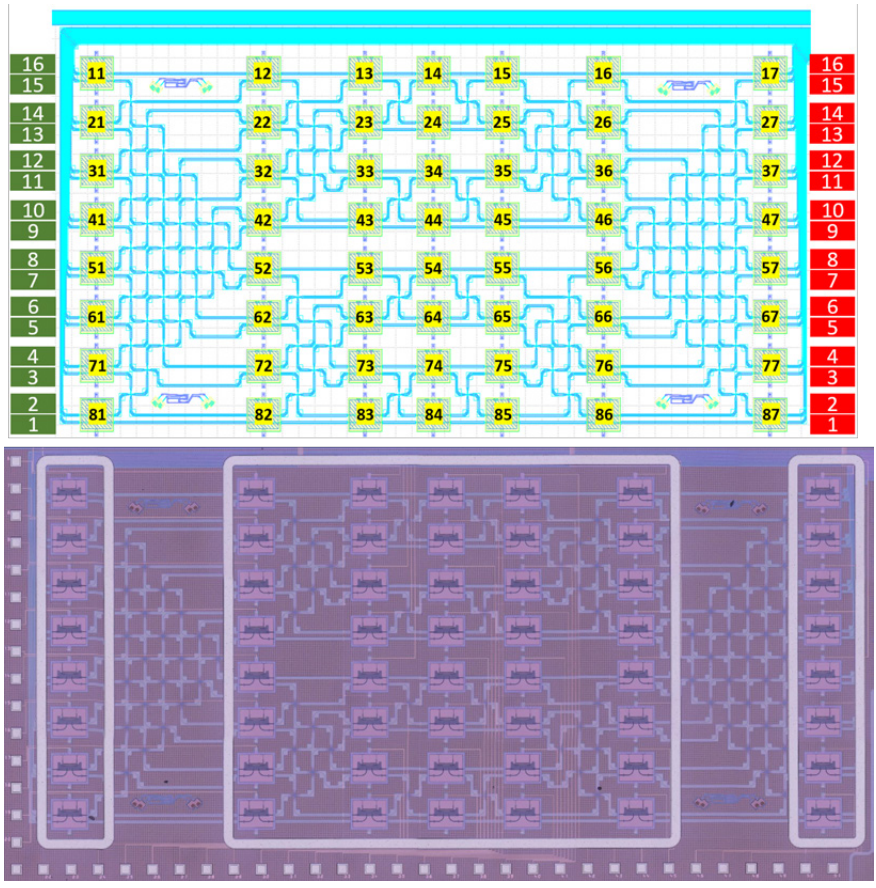


Figure 6.37: Benes 16×16 switch circuit on MORPHIC RUN2 chip.

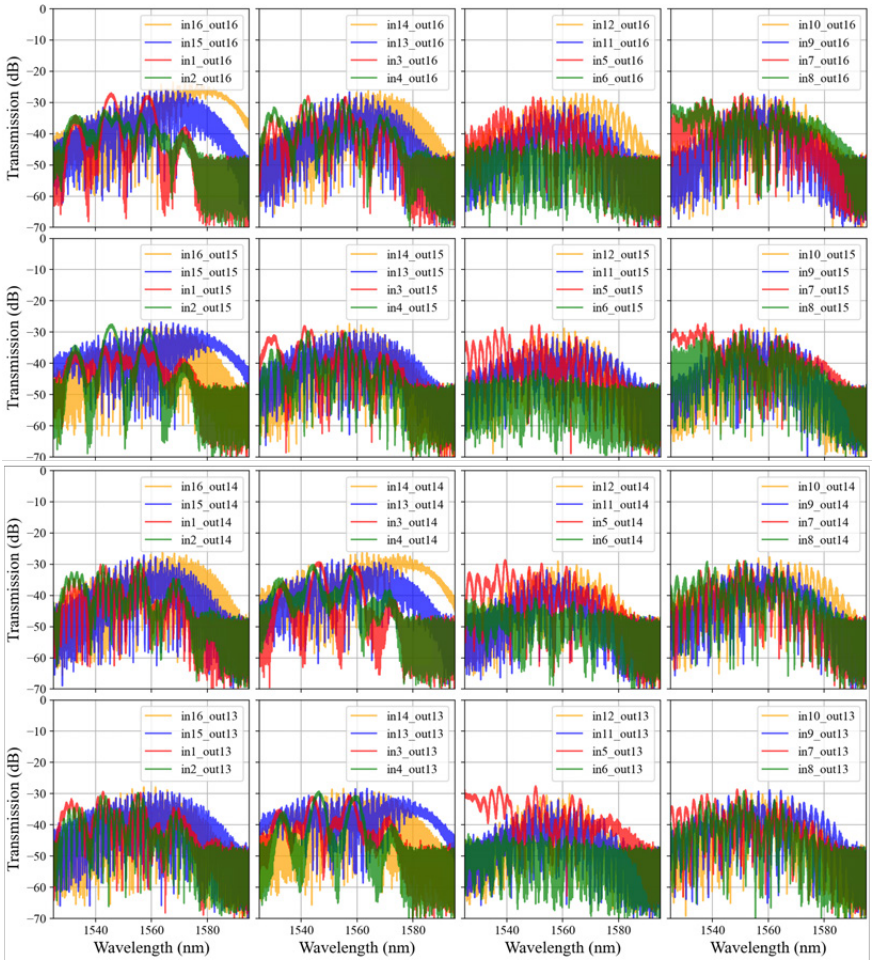


Figure 6.38: Transmission measurements of the Benes 16×16 switch circuit on an unprocessed RUN2 chip for various input and output combinations: input ports (1-16) and output ports (13-16).

6.3.5.1 Unprocessed Chip

Figure 6.38 shows the transmission measurements of the Benes 16×16 circuit on an unprocessed RUN2 chip for all the inputs and outputs from $out16$ to $out9$. Since the circuit layout is symmetric, the grids have been constructed in a way that transmission responses of the mirrored couplers are next to each other. For example, $in16$ and $in15$ of the coupler 11 in the first row are mirrored by $in1$ and $in2$ of the coupler 81. As expected, the loss of this circuit is higher than the previous circuits due to the larger number of switches (7) in each path. And, various interference

patterns can be seen. The highest transmission is related to the $in16 - out16$ pair and is about $-25\,\mathrm{dB}$.

Similar to the Benes 4×4 circuit, we observe transmissions with varying FSRs, indicating large and small path-length differences for each input-output pair. As an example, the transmission responses of the $in1 - out16$ and $in16 - out16$ are plotted in Fig. 6.39a, which implies the higher number of possible paths for the $in16 - out16$. This can be seen in the schematics shown in Fig. 6.39c and Fig. 6.39d.

Additionally, the transmitted powers in the possible paths of $in1 - out16$ and $in16 - out16$ pairs using inverse Fourier analysis of the Luna OVA have been illustrated in Fig. 6.39b. The first pair has diagonal connection while the other one has the simplest connection of the circuit. Compared to the $in16 - out16$ pair, the diagonal connection of the $in1 - out16$ ports causes has fewer paths contributing to the light interference. This is clearly seen in the Fig. 6.39d where number of peaks for $in16 - out16$ pair goes up to 8, while the other one has only 4 peaks.

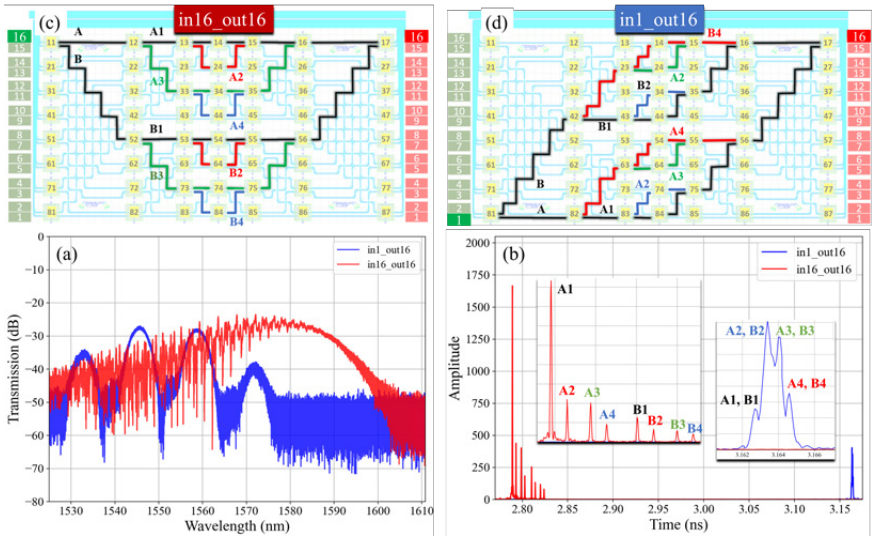


Figure 6.39: Identification of the transmitted powers in possible paths using the inverse Fourier analysis of the Luna OVA for Benes 16×16 switch circuit on an unprocessed chip. (a) $in1$ to $out16$. (b) $in16$ to $out16$. Schematics show all the possible light paths for the selected input and output.

Figure 6.40 shows the grid for the reflection measurement of the Benes 16×16 circuit for input ports of 1-16 and output ports of 9-16. Compared to other circuits, the reflection peaks of the output grating couplers are not clear in the plots. This is because the light is now distributed over many output grating couplers at different

distances, so the reflection peak is not completely smeared out. We also see an increase in the number of peaks within the circuit, which is obvious because there are many more couplers and crossings in the circuit. Figure 6.40 clearly shows two types of reflection peaks: the narrower peaks correspond to the crossings while the broader ones can be attributed to the couplers.

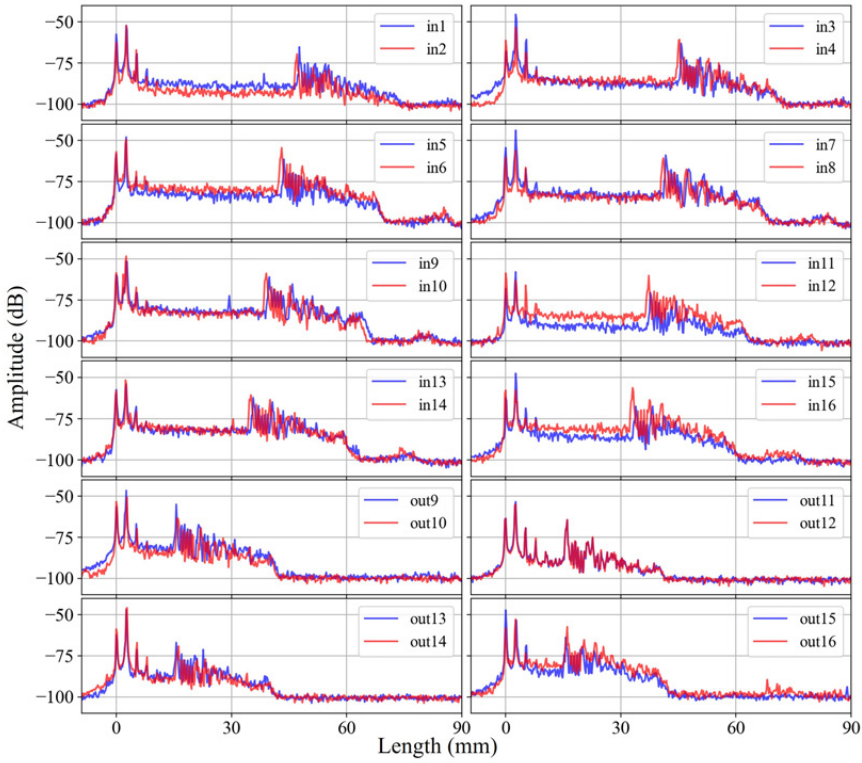


Figure 6.40: a) Reflection measurements on the Benes 16×16 switch circuit on an unprocessed chip. From the reflection impulse response, we can see the reflections of the grating couplers, switches and the output grating couplers. As seen, there are two types of reflection peaks the narrower peaks correspond to the crossing while the broader ones can be attributed to the couplers.

6.3.5.2 Packaged Chip: Full Demonstrator

Our transmission measurements show no power transmitted for any of the port combinations. As an example, the transmission response for the $in16 - out16$ pair is shown in Fig. 6.41. This indicates that enough couplers have collapsed to block all the possible paths through the circuit. It is most likely that in many cases only the North arms have collapsed, but as there is also no transmission from $in1$ to $out1$ (the Southmost path with only rigid arms) there is indication that also some South arms might have broken.

The collapse of the MEMS is also confirmed by looking at the reflection from $in1$ in Fig. 6.42: we see only a single reflection peak corresponding to the first coupler. In none of the reflection plots can we see the expected seven peaks related to couplers, there are usually only one/two peaks related to the couplers and rest belong to the waveguide crossings. Furthermore, there is no indication of the output grating couplers at all.

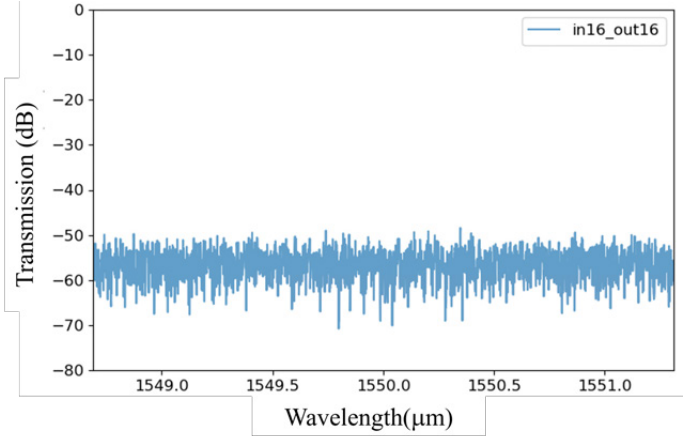


Figure 6.41: Transmission measurement of the Benes 16×16 switch circuit on the full demonstrator chip for $in16$ and $out16$. Other port combinations have the similar flat noise response.

As mentioned before, the full demonstrator contains both small-scale and larger switches designed on RUN2. The results are not encouraging: it is seen that many of the channels through the circuits show very low or zero transmission, indicating a collapse of many MEMS devices. In the situations where a decent transmission is not seen, such as with the Crossbar 4×4, first actuation tests indicate that the couplers that are not collapsed are bridged and cannot be actuated. With the combination of both transmission and reflection measurements we should be able to get a good picture of exactly which paths in the different circuits have survived,

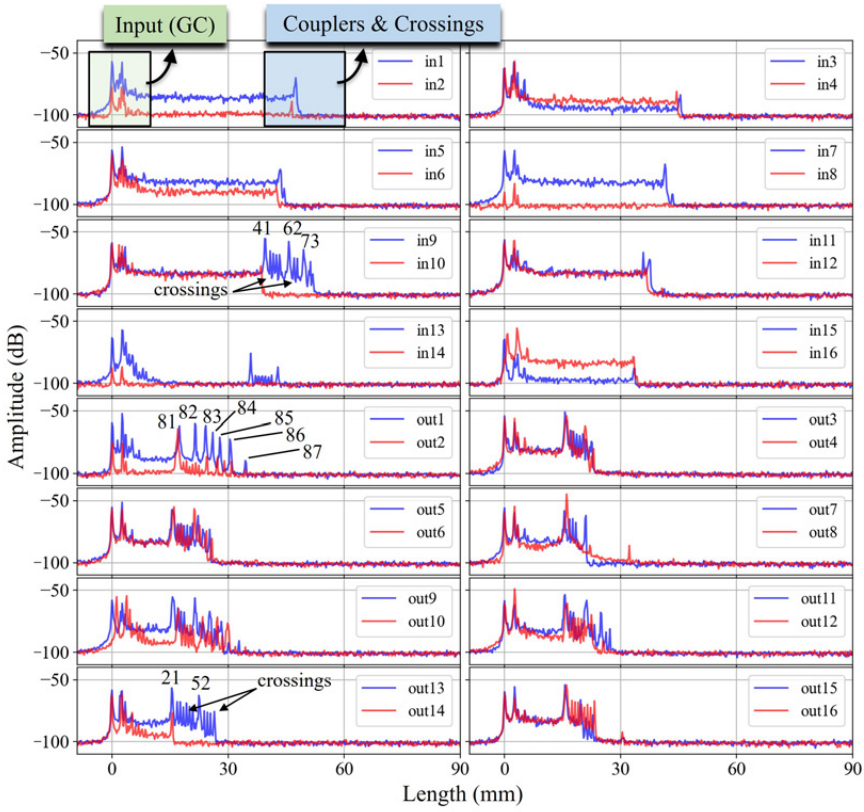


Figure 6.42: Reflection measurements on the Benes 16×16 switch circuit on the full demonstrator chip.

and further actuation tests could teach us more about the actual state of the MEMS devices. Actuation measurements are discussed in Sec. 6.4

6.3.5.3 Packaged Chip: Mini Demonstrator

The situation does not improve with the larger Benes 16×16 circuit on the mini demonstrator (version 2). Just like with the full demonstrator, this circuit has no transmission response even for the simple paths of $in1 - out1$ and $in16 - out16$ pairs (Fig. 6.43).

Figure 6.44 shows the reflection measurements for the Benes 16×16 switch circuit on the mini demonstrator (version 2) chip. There are several points to mention about this grid of results:

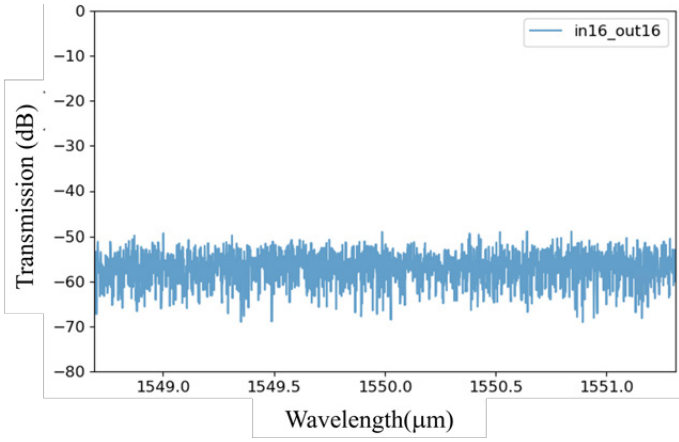


Figure 6.43: Transmission measurement of the Benes 16 × 16 switch circuit on the mini demonstrator (version 2) chip for in16 and out16. Other port combinations have a similar flat noise response.

1) Single peaks in the coupler region shows that the first coupler has collapsed.

2) There is no sign of the peaks for the output grating couplers.

3) In some couplers the flexible movable arm has collapsed but the rigid fixed arm is still suspended, and light can pass through it. For example, $out2$ is connected to the fixed arm, while $out1$ is connected to the suspended arm. If we check their measured reflections with the circuit layout, we can see that the number of crossings matches the number of the peaks for $out2$. On the other hand, $out1$ shows only one sharp peak.

The mini demonstrator (which was actually fabricated before the full demonstrator) shows even more disappointing results in the transmission and reflection measurements. While some basic actuation tests were done on the few devices that showed some light coming through, this did not yield any response. Everything indicates that most of the MEMS devices have collapsed, or at least the flexible movable parts of the couplers, while the rigid fixed waveguide part remains suspended.

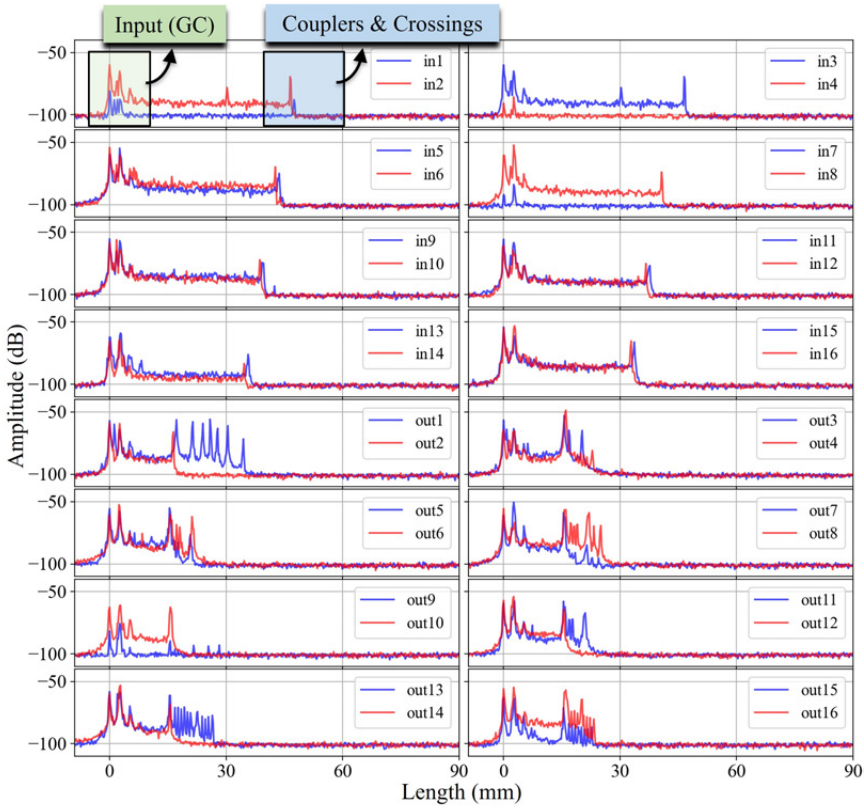


Figure 6.44: Reflection measurements on the Benes 16×16 switch circuit on the mini demonstrator (version 2) chip.

6.4 Actuation Test

As discussed in the previous sections, passive measurements of the packaged demonstrators show that all the circuits have failed and there are no operational circuits. For further investigation, I tried to actuate the couplers on the (apparently) healthy routes with light transmission. This was done for different circuits, however no considerable change in transmission was observed. As an example of these measurements, the Crossbar 4×4 switch circuit on the full demonstrator is selected to be presented here. Moreover, we also present the actuation test for the test nodes on RUN3 for one of the sealed chips. Even the test nodes had failed due to the collapse of the MEMS actuators.

6.4.1 Crossbar 4×4 (Packaged Chip: Full Demonstrator)

The crossbar 4×4 on the full demonstrator chip has the best transmission among all the measured circuits. And, based on its results it is expected that coupler 41 of the circuit should work properly (Fig. 6.45). Therefore, the actuation measurement is performed on this coupler to see whether it moves or is bridged. Figure 6.45 shows the transmission response of the Crossbar 4×4 for the different actuation voltages of 0, 5,10, 15, 20 volts applied to the coupler 41. As seen, the transmission response is affected. Also, by applying voltage (0 to 5 and 15 to 20) a sudden shift in transmission response of the circuit is seen which is not exactly our desired outcome. For this vertically actuated MEMS device, it is expected to see a small change at small voltages (maybe even a small improvement) but then a rapid drop in transmission to port $out1$ as the two waveguides move away from each other. Instead, a strong change for a low voltage is seen, then a very small change, and finally for even higher voltages, suddenly a larger change is observed again.

Also, we see the steps for the 20 V actuation (purple curve) which suggest that something has happened during the measurements (we could not identify it). An abrupt change in wavelength response does not indicate a wavelength dependence, but a change over time, as the wavelength response is collected as a sweep over time taking several seconds to complete.

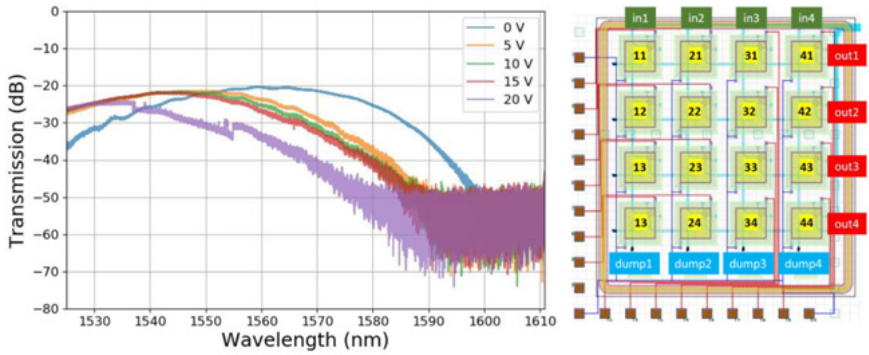


Figure 6.45: Transmission measurements of the Crossbar 4×4 switch circuit on the full demonstrator chip for the light going from $in4$ to out1 for actuation voltages of 0, 5, 10, 15, 20 volts.

6.4.2 Test Nodes (Sealed Chip)

Apart from the demonstrator circuits, we also characterized additional test circuits. These circuits are generally smaller in scale and they are not connected to either

of the 72 fiber arrays. As a result, they do not require the full packaging flow for testing, but generally go through a simple, customized packaging using wirebonding. As these chips do not require an interposer and thus are not fully encapsulated, they can be inspected in different ways, providing insight in the failure mechanisms for the photonic MEMS.

For further inspection of the MEMS collapse, we also performed the actuation measurements for the test nodes for more details) on the etched and sealed chips. As shown in Fig. 6.46, the test nodes circuit includes two nodes: the basic node (three couplers) and the full node (three couplers and three phase shifters connecting the couplers).

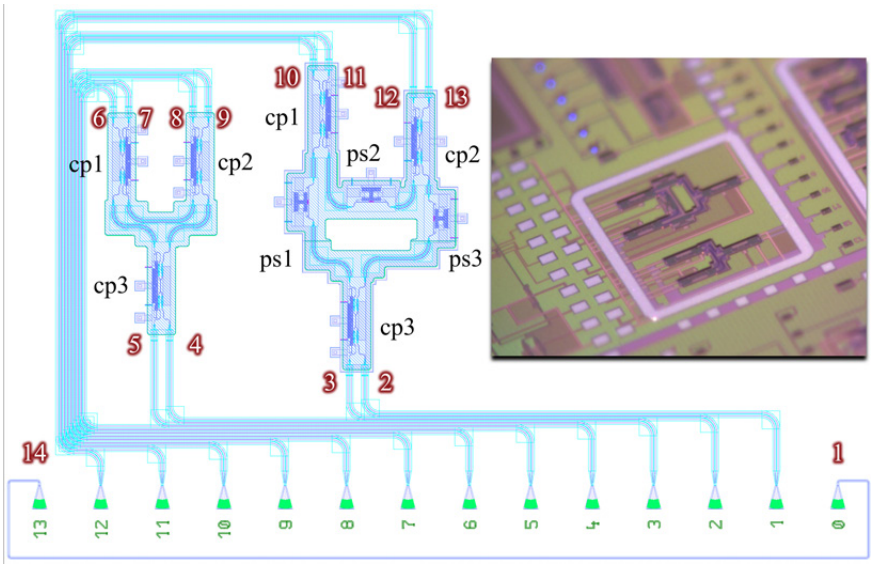


Figure 6.46: Test nodes circuit on MORPHIC RUN3. Layout and the image of the circuit on an unprocessed chip (for visibility of the nodes, we did not use the sealed circuit image).

The measurement setup is similar to the passive measurement of the unprocessed chips. And, for the MEMS actuation, three DC probes have been used: chip ground (Fig. 6.47d), circuit ground (Fig. 6.47c), and the one for the signal pads (Fig. 6.47c-d). Chip and circuit grounds are both connected to the voltage source ground. For the optical measurements, an array of 14 grating couplers have been located on the South part of the circuits which supports a shunt waveguide and 12 ports to access the test nodes.

We characterized the test nodes on various sealed and etched chips. Unfortunately, in all cases, the MEMS had already collapsed or collapsed during the measurement. Here, we demonstrate one of our measurements results on a sealed

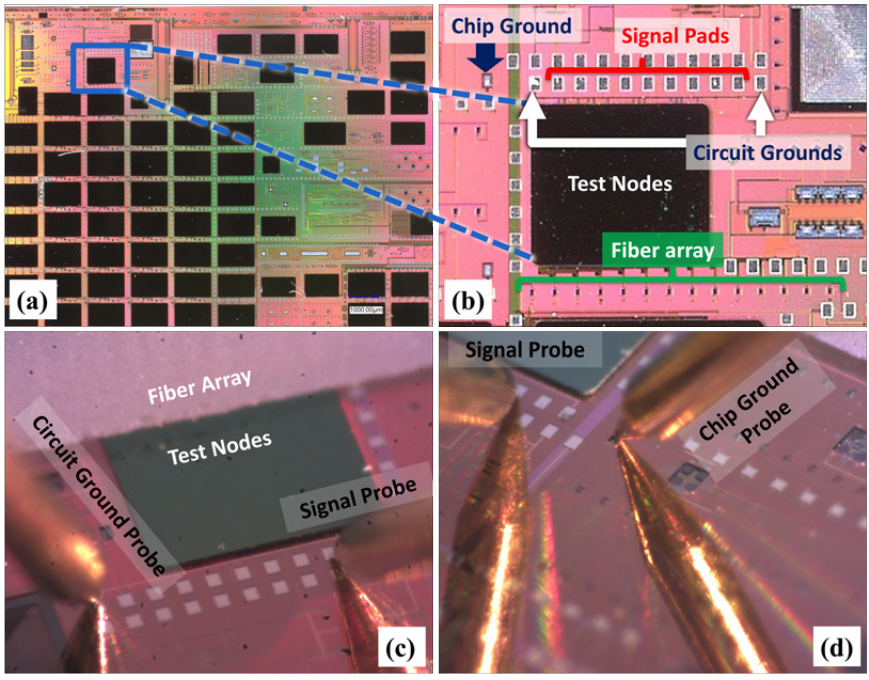


Figure 6.47: a) Location of the test nodes circuit on a RUN3 sealed chip, b) Position of the electrical pads and optical ports of the circuit, c-d) demonstration of DC probes connection to the chip ground, circuit ground, and signal pads.

chip showing the collapse of the MEMS. The measurements consist of 5 steps:

- Step 1: Connecting the ground probes to the substrate grounder and ground bondpads. And, measuring optical transmissions for all the ports pairs.
- Step 2: Connecting the signal probe while keeping keithley voltage at $0\,\mathrm{V}$. And, measuring optical transmissions for all the ports pairs.
- Step 3: Start sweeping voltage from $0\,\mathrm{V}$ to $10\,\mathrm{V}$, and measuring optical transmissions for the the ports pairs related to the selected coupler. For example, pair 4_6 for coupler 3 in the basic node.
- Step 4: Measuring optical transmissions for all the ports pairs, when the actuation of the selected coupler is done.
- Step 5: Lifting all the probes and measuring the optical transmissions for all the ports pairs.

Figure 6.48 summarizes the transmission responses of the all ports pairs (in one of the sealed chips) for measurement steps of 1, 2, 4, and 5. As seen, after completing the actuation cycle (step3) most of the actuators have collapsed in transmission measurements of step 4. And, after lifting the probes in step 5, the remained actuators also collapsed.

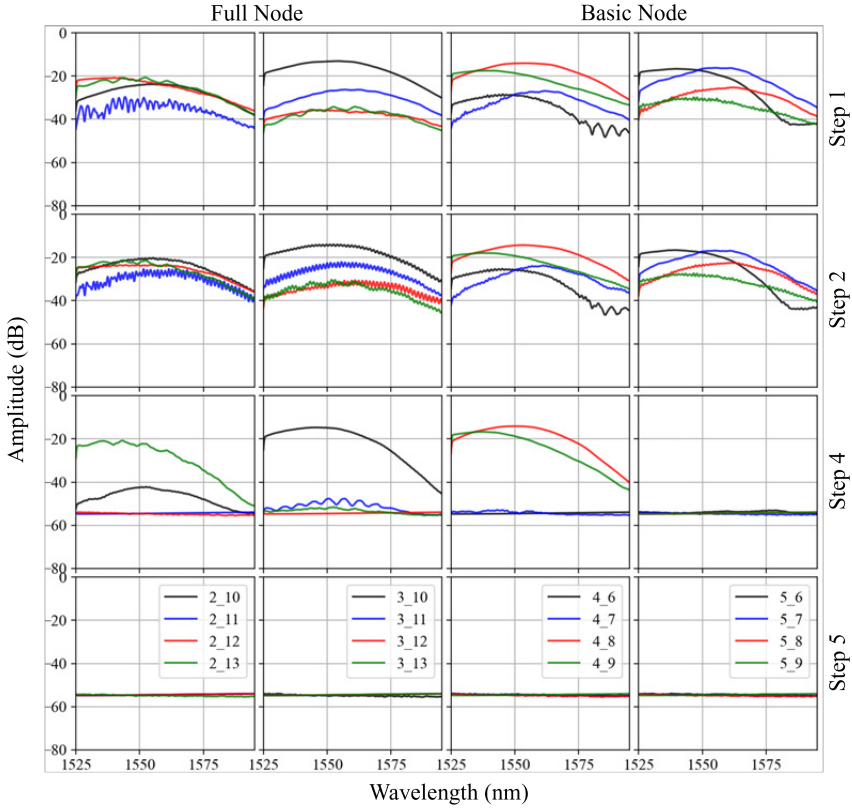


Figure 6.48: Transmission response of all the ports pairs of the basic and full nodes in the test nodes circuit on a MORPHIC RUN3 sealed chip. Step 1: After connecting the ground probes. Step 2: After connecting the signal probe to the coupler 3 of the basic node. Step 4: After completing the actuation cycle in the step 3. Step 5: After lifting all the probes.

In figure 6.49, we have shown the transmission responses of the ports pair $4-6$ during the actuation cycle (step 3) of the coupler 3 in the basic node. As seen, after increasing the voltage from $0.5\,\mathrm{V}$ to $1.0\,\mathrm{V}$, the coupler suddenly collapses and the transmission drops to $-50\,\mathrm{dB}$.

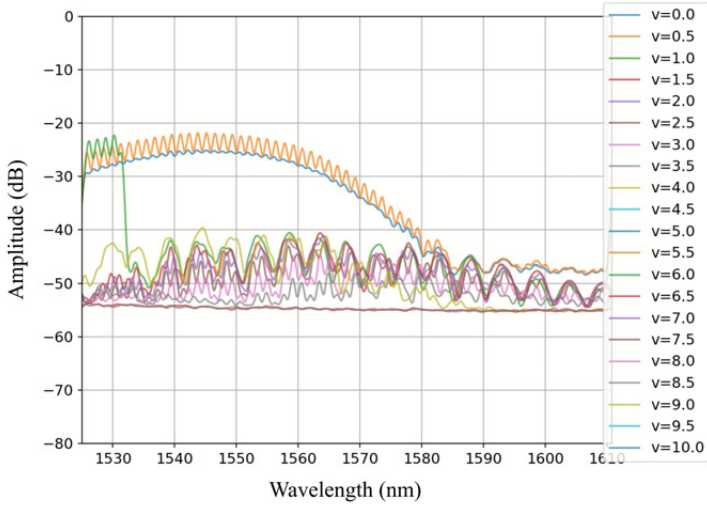


Figure 6.49: Transmission responses of the ports pair $4-6$ *during the actuation cycle (step 3) of the coupler 3 in the basic node. We accidentally could capture the collapse of the coupler at actuation voltage of* $1.0\,\mathrm{V}$ *(green curve).*

6.5 Failure Analysis

Although individual MEMS devices have a high yield, it is surprising to find that most MEMS devices have collapsed in the packaged demonstrator circuits. Understanding which process steps and mechanisms cause this collapse and how to prevent it in future demonstrators is crucial. Significant effort has been dedicated to developing new inspection methods and tests to determine the extent of the failure and identify the root cause of the issue. This section briefly discusses the results of three such investigations[5]:

1) The disassembly of a sealed device that had collapsed after a simple wire-bonding procedure.

2) Test different wirebonding process parameters to check if the ultrasonic welding can cause a collapse .

3) Infrared camera inspections through the sealing lids to establish the yield of the devices after sealing .

[5]I should mention that these analyses were conducted by others in the group and within the MORPHIC project. However, I discuss them here briefly to help the reader understand where issues arose. The experiments in steps 1 and 2 were mostly done by Dr. Umar Khan and Steven Verstruyft at UGent. And, step 3 was conducted at KTH by Prof. Kristinn Gylfason.

6.5.1 Inspection of Sealed Devices

Fully packaging and encapsulating of the MEMS circuits on the demonstrators chips makes the inspection processes (removing the lids) challenging. Hence, the inspection is performed on a small-scale MEMS-based 4×4 beamformer on the MORPHIC RUN3 chip (Fig. 6.50a). This circuit had a simple packaging using wire-bonds (Fig. 6.50b) for connecting its MEMS devices to the EIC board.

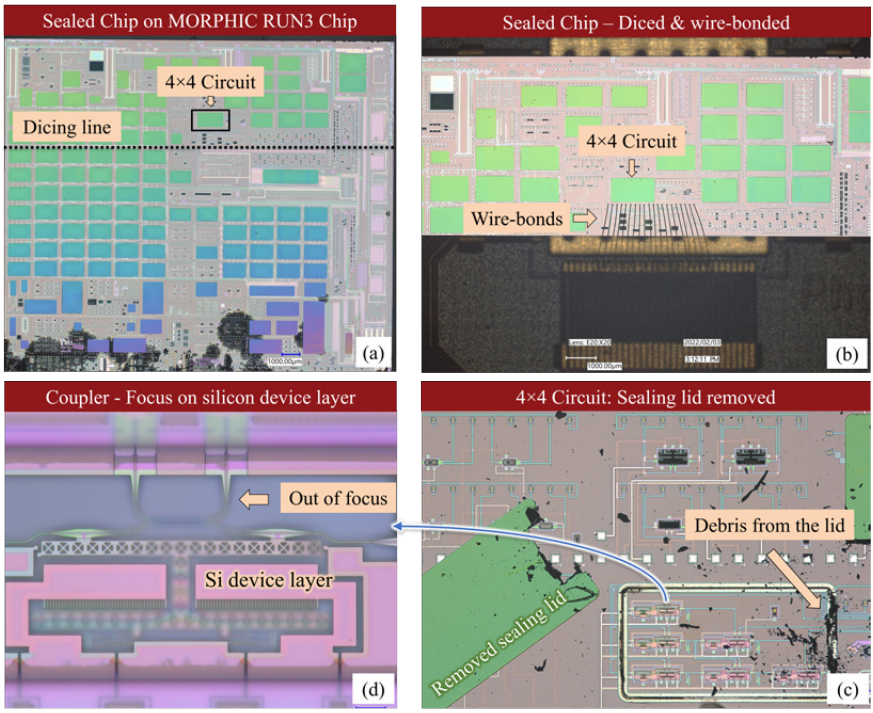


Figure 6.50: Disassembly of a sealed 4×4 beamcoupler circuit on a MORPHIC RUN3 chip. (a) Position of the circuit on the chip and the dicing line. (b) Diced and Wirebonded circuit. (c) 4×4 circuit after removing the lid. (d) Zoom with small depth of focus. The defocus shows that the waveguides have collapsed. (Image Credit: Dr. U. Khan)

Transmission measurements across all port combinations revealed that no light reached the output from any selected input. Even the shortest route, which involved only a single coupler, showed no transmission. For the inspection, the 4×4 circuit was disassembled. The lid from the circuit was mechanically pried off using tweezers and the actuators were inspected under a microscope as shown in Fig. 6.50d. The color of the suspended waveguides indicates that they have collapsed: when the waveguide touches the silicon, it becomes dark. And when there is a strong variation in the distance of the waveguide from the substrate, Newton rings can be

observed. Also, focus of the microscope with high magnification (consequently small depth of focus) can be used to estimate the displacement. Performing this inspection on all 15 actuators in the circuit confirms that everything has collapsed. Again, the origin of the problem is unknown, and is probably not exactly the same as for the packaged demonstrators, as different processes were used.

6.5.2 Wire-Bonding Experiments

One potential explanation for the collapse is the occurrence of electrostatic discharge (ESD) or ultrasonic vibrations during the packaging process. These phenomena can manifest while executing procedures like wirebonding and stud-bumping, which are used in the different packaging flows. In an effort to deepen our understanding, wire-bonding experiments were conducted in UGent using varying ultrasonic parameters. The findings from these tests confirm that wirebonding can indeed trigger collapse, although the degree of impact varies across different settings. While the results are not yet definitive, they do offer insights into potential collapse mechanisms. By examining instances of device or circuit collapse in relation to their mechanical proximity and electrical connections to the pads being bonded, it becomes possible to infer distinct collapse mechanisms.

For the wire-bonding test carried out in UGent cleanroom, a MPP iBond 5000 wedge wire-bonder is used. Important controlling parameters of the wire-bonder are power (p), time (t), and force (F) with min-max ranges of $1.3\,\mathrm{W}$-$2.5\,\mathrm{W}$, $10\,\mathrm{ms}$-$1000\,\mathrm{ms}$, and $10\,\mathrm{g}$-$250\,\mathrm{g}$, respectively. Several iterations and parameter sweeps were performed to find out the minimum values required for a strong bond between the wire and the bond-pad. After each wire-bonding experiment the device under test was inspected through a microscope to see whether the MEMS have collapsed or not. For these tests, starting values in arbitrary unit (a.u.) were $p = 0.52$, $t = 3$, $F = 3$, and final values turned out to be $p = 2$, $t = 6$, $F = 3$ which were used for the final tests.

Note that these experiments were performed with unsealed samples, where the suspended MEMS are surrounded by air. The sealed MEMS are suspended in a vacuum. Therefore, they will experience less mechanical dampening, and might be more prone to collapse by an abrupt ESD-like spike.

Figures 6.51 and 6.52, show the microscope image of the two identical phase shifters used in the wire-bonding test, and as seen, shadows of the waveguides are clearly observable for both cases before wire-bonding (Fig. 6.51 and Figs. 6.52a,b). For the first structure, two wire-bonds were used simultaneously and as seen after wire-bonding the substrate grounder has collapsed (Fig. 6.51b). For the second phase shifter, only one wire-bond was used for the test. First, the wire-bond

was attached to the grounder bond-pad and after the attachment no collapse was observed (Fig. 6.52c). However, attaching the second wire-bond it resulted in the collapse of not only the phase shifter but also the substrate grounder which had stayed intact after the first wire-bond.

To conclude the wire-bonding tests, different experiments have been conducted to figure out whether wire-bonding is detrimental for MEMS devices or not. The initial tests suggest that wire-bonding should be avoided for MEMS devices. However, it should also be mentioned here that substrate grounder did not collapse after the wire-bonding of the bond-pad connected to it. And the main cause of the collapse is related to the phase shifter itself. Despite of these experiments, further investigations are needed to figure out if wire-bonding is really detrimental to the MEMS devices.

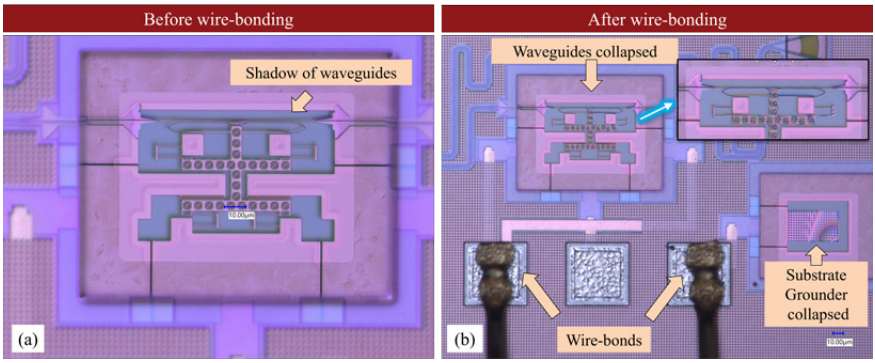


Figure 6.51: (a) Microscope image of a MEMS phase shifter prior to the wire-bonding tests. The suspended parts of the phase shifter are intact, as indicated by the visible shadow, and the waveguides match the color of the silicon device layer. (b) After wire-bonding, both the substrate grounder and the phase shifter have collapsed. (Image Credit: Dr. U. Khan)

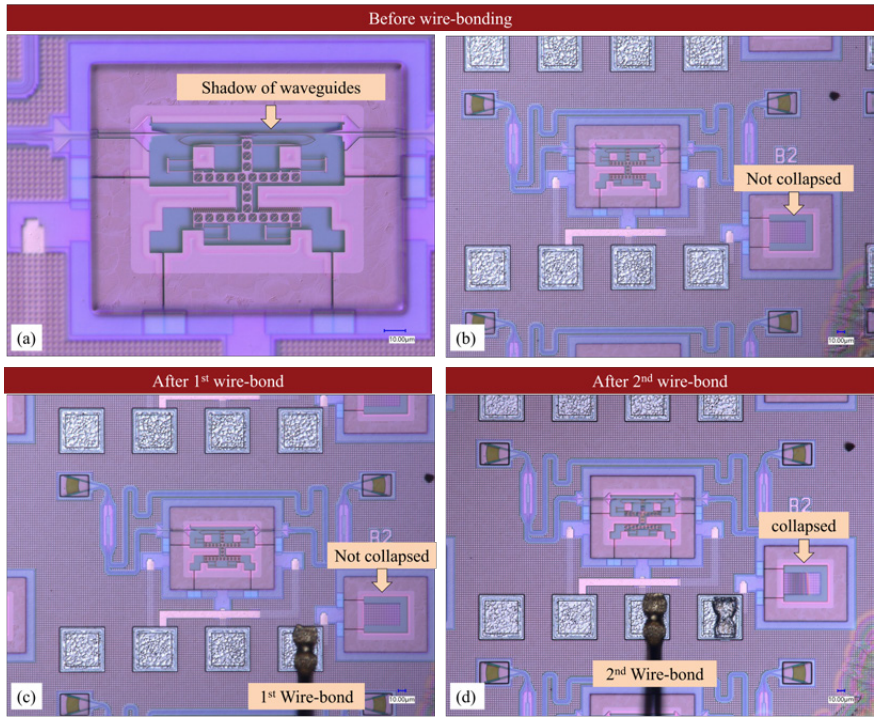


Figure 6.52: (a) Microscope image of an intact MEMS phase shifter, confirmed by the visible shadow and matching color of the suspended parts. (b) Microscope image showing the substrate grounder next to the phase shifter in a suspended state. (c) The substrate grounder remains intact after wire-bonding the electrical bond pad connected to it. (d) Both the phase shifter and the grounder collapsed following the second wire bond. (Image Credit: Dr. U. Khan)

6.5.3 Infrared Camera Inspections

Another method of inspection is using cameras sensitive to the infrared wavelengths. Since silicon is transparent to these wavelengths, it is possible to image through the sealing lids made of silicon. Hence, the sealing lids designed in RUN3 have openings in the gold layer to enable imaging in the infrared wavelength range. For this, a TELOPS FAST M35 camera was used which is sensitive to light in the $3\,\mu\text{m}$-$5\,\mu\text{m}$ spectral range. These tests were done at KTH where they have a thermal inspection setup including this camera and a motorized stage that makes it possible to collect many images from the same chip in an automated way.

Since the smallest device dimensions are well below the diffraction limit of a camera operating in the $3\,\mu\text{m}$-$5\,\mu\text{m}$ spectral range, individual waveguides in the images cannot be resolved. However, suspended and collapsed MEMS devices can be distinguished, as the actuators have a footprint of a few microns. Since collapse to the substrate is a major failure mode of the photonic MEMS devices, this information is valuable. The contrast in the images is likely due to an increase in transmission when the silicon device layer collapses to the surface since the collapse removes two high contrast interfaces and their associated Fresnel reflections. Thus, a brighter region in a suspended waveguide indicates a collapsed section. Also, in the out-of plane curved sections connecting collapsed and suspended device layer regions, periodic interference effects (Newton rings) can be seen. This is particularly clear for the substrate grounder devices that are purposefully made to collapse.

Figure 6.53 shows two examples of infrared images of the standard unit cell of the large 126-cells FP-PIC designed on RUN3 discussed in Chapter 3. Each such standard unit cell contains 6 couplers and 6 phase shifters. Figure 6.53a is an example of a unit cell with low yield, since all the phase shifters and couplers have bright spots in their suspended regions. Figure 6.53b, however, is an example of a unit cell with high yield since no bright spots are visible in the suspended waveguide segments in the MEMS devices. Here, it is also seen that the substrate grounder has collapsed, as intended, since the suspended section is brighter than the supported part.

This test was carried out for the 124-cell FP-PIC circuit on five different chips from RUN3, where one chip had a severe etch attack and the other one had stud-bumping process. Results show that couplers display a more variable yield than the phase shifters. The phase shifters seem either to all be suspended (the majority) or all be collapsed, within each cell. Upon closer inspection, it became clear that most cells displaying catastrophic failure have damage to the surface of the back-end-of-line (BEOL) oxide, due to break-through (“etch attack”) of the HF

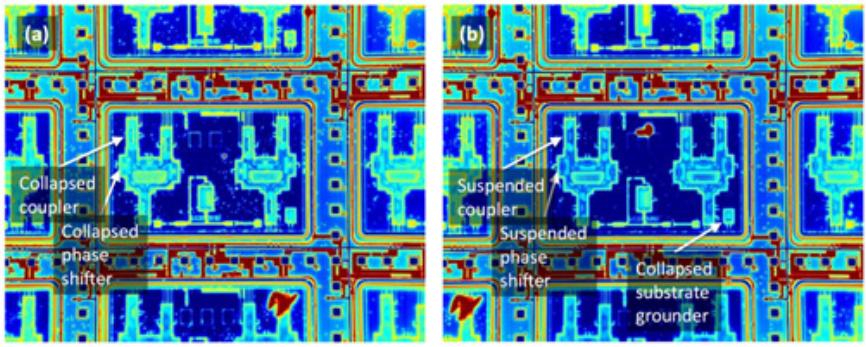


Figure 6.53: Example mid-IR images collected through the sealing lids of the unit cells of the large FP-PIC demonstrator on RUN3. (a) An example of a unit cell with a low yield. All the 6 couplers and all the 6 phase shifters of the unit cell have collapsed. (b) An example of a unit cell with high yield. All the couplers and phase shifters are suspended, while the substrate grounder is collapsed, as intended. (Image Credit: Prof. K. Gylfason)

vapor release etch through the alumina passivation layer. This results in a failure of the lid sealing process and leakage into the MEMS cavity during the subsequent processing steps. Interestingly, the chip that underwent stud-bumping does not deviate from the others, indicating that the stud-bumping process is not negatively impacting yield.

The total suspension yield results over all cells on the five chips are summarized in Table 6.2. The phase shifters have a significantly higher suspension yield than the couplers. This is likely due to the optimized mechanical design of the 3rd generation phase shifter used in RUN3. The couplers, however, were a first-generation design. For chips that do not suffer severe etch attack, the phase shifter suspension yield is better than 90%, while the couplers reach 78% outside areas of etch attack on the best chip.

Table 6.2: Suspension yield, i.e. the fraction of MEMS devices that appear suspended, of the standard MEMS couplers and phase shifters used in the MORPHIC RUN3 FP-PIC demonstrator circuit, as determined by IR inspection through the sealing lids of five sealed chips. One of the chips had also undergone stud-bumping.

Chip	CPs	CPs outside etch attack	PSs	PSs outside etch attack	Note
1	0.42	0.71	0.44	0.67	Severe etch attack
2	0.69	0.78	0.79	0.89	
3	0.45	0.55	0.98	1.00	
4	0.44	0.44	0.95	0.95	Stud-bumped chip
5	0.69	0.82	0.79	0.98	

In summary, infrared inspection through the sealing lids of the MEMS cavities is a powerful method for evaluating the suspension yield of the MEMS devices. Future work in this direction should establish the correlation between the observed suspension and actual device functioning. Also, for large scale yield analysis, automated image analysis would be helpful to speed up the process.

6.5.4 Further Measures

There are also additional measures to test the failure modes of the MEMS devices, or to reduce the chance that they occur.

- **Stiffer MEMS designs**: Increasing the thickness of the silicon device layer directly enhances bending stiffness, allowing for the fabrication of more robust MEMS devices. However, this becomes challenging if the waveguide silicon layer thickness cannot be changed.
- **Substrate grounding**: The SOI substrate is encapsulated by dielectric materials. For instance, the backside is covered by a thick oxide and nitride film to compensate for stresses in the top layers and reduce wafer bow. These backside layers could be removed or opened (or the substrate could be thinned down) after which it could be metallized to provide a good grounding with a reliable potential. Use of low-resistivity substrates is also possible but this would probably degrade the performance of the high-speed modulators.
- **ESD protection**: The iSiPP50G platform does provide the capability to include diodes in the photonic layer (this is used to make modulators). These diodes could be incorporated in the bondpads to provide ESD protection for the MEMS devices. This is commonly done in electronics, but not usual for photonic devices.
- **Alternative wirebonding / stud bumping methods**: There exist processes that use laser welding to attach a wire or stud-bump to a pad, and this process does not induce ultrasonic vibrations.

6.6 Conclusion

The testing of the demonstrators in this chapter provides an insight into the performance and challenges associated with the MORPHIC platform's photonic MEMS-based circuits. While the project successfully designed and assembled several demonstrator circuits, the testing phase revealed significant issues, primarily the

widespread collapse of MEMS actuators. This collapse, as identified through passive and active measurements, led to a drastic reduction in optical transmission and rendered many circuits non-functional. The findings suggest that critical steps in the packaging process, such as electrostatic discharge, ultrasonic vibrations during wire bonding, or substrate grounding, may have contributed to these failures. Despite the challenges, the detailed characterization using advanced tools like optical vector analysis (OVA) and infrared camera inspections has allowed us to identify potential failure mechanisms, guiding future improvements.

The results of the full demonstrators indicate that while some circuits demonstrated partial functionality, many MEMS devices either collapsed or were bridged, preventing successful actuation. This was further confirmed through reflection measurements and initial actuation tests. Similarly, the mini demonstrators tested also exhibited a high rate of MEMS collapse, compounding the difficulties faced during the project. However, the tests did provide useful data for understanding the nature of MEMS failures, which will inform the future optimization of the packaging flow.

Looking ahead, packaging processes need to be refined to prevent MEMS collapse in the next demonstrators. Additional debugging and characterization work will be required to pinpoint the exact cause of the failures. By addressing these challenges and implementing corrective measures, the MORPHIC platform has the potential to achieve its goal of creating robust, large-scale photonic circuits based on reconfigurable MEMS technology. The lessons learned from these initial tests will play a critical role in overcoming the current limitations and driving further innovation in the field of programmable photonic integrated circuits.

7

Conclusion and Perspectives

The objective of this thesis was to leverage the MEMS-enhanced silicon photonics technology to create a new class of programmable photonic circuits, large-scale generic Field-Programmable Photonic Integrated Circuits (FP-PICs), that can be used for a variety of functions and to be programmed in software. The presented results and discussions in this work are part of the European Horizon 2020 project MORPHIC (MEMS-based zerO-power Reconfigurable Photonic ICs).

In our approach, realization of MEMS-based programmable circuits requires several major steps of:

- Design and optimization of the optical building blocks such as couplers and phase shifters: The main objective of this step is to introduce actuators with low power consumption, compact size, low insertion loss, and short response time.
- Design of the optical core: A recirculating mesh (consist of the phase shifters and couplers connected by the waveguides) with proper topology, shape, and size. This core should also be connected to the other components such as active devices (e.g. photo detectors and modulators) and to the specialized units such as a phased array antenna, delay lines, and so on. Another important aspect of the mesh design is proper implementation of the monitors within the mesh to provide feedback for the control and stabilization of the

configured mesh.

- Design of the packaging interfaces (interposers and PCB interconnects) to connect the photonic chip to the electronic drivers.
- Chip fabrication and post-processing: This includes the development of a MEMS release process that is compatible with an established foundry platform. We also need to implement the hermetic sealing to protect MEMS devices from environmental influences.
- Packaging: The main goal of the packaging is to make sure that both optical and electrical signals can traverse various interfaces between the photonic chip and the external hardware such as light sources and controlling electronics. In fact, it integrates all components into a single physical system and links it to the external environment through optical, electrical, and high-speed connectors.
- System integration: To build a programmable photonic chip controlled by software, we are actually making a system with different physical and abstract layers where all the components are somehow connected together and a small change in one component can effect other components or even the entire system. For example, geometry of the MEMS can affect the choice of DACs as our voltage drivers or the distance of the pads on the bondpad grid can determine required technology to fabricated interposers.
- Implementation of a software framework: To enable the efficient design, configuration, control, and optimization of programmable circuits. Since FP-PICs involve complex, reconfigurable photonic hardware, a software framework simplifies interaction with the underlying technology and enhances programmability, scalability, and performance.

In MORPHIC, EPFL and KTH were responsible to design and develop the MEMS-actuators. For photonic chip fabrication, the consortium used the standard IMEC silicon photonics platform iSiPP50G, which has passive waveguides in 3 etch depths into a $220\,\mathrm{nm}$ silicon-on-insulator (SOI) stack. It also has carrier-based modulators, germanium photodetectors, and 2 layers of metal interconnects and a final passivation/bondpad stack. And, the post processing steps were carried out in EPFL and KTH. The Packaging process, design of the electronic drivers, multilayer interposers and PCBs, and low-level API drivers was done by Tyndall.

My contribution in MORPHIC was to design the optical core (mesh) and consequently the circuit of the FP-PICs, and, also, to design various single-layer interposers and PCBs for mini demonstrators (FP-PICs and switch circuits). Additionally, I developed a software framework facilitating the design, analysis, and

control of the FP-PICs. Moreover, I was involved in generation of the netlists which is an important step in packing and assembly process as well as mesh configuration and circuit programming.

7.1 Parasitics

For parasitic analysis we focused on the hexagonal meshes and studied various configurations for the 7-cell hexagonal mesh. The source of parasitics can be issues such as inadequate fabrication process or discretization error of the DACs. We showed that by putting the unused couplers of the mesh in cross state we can considerably reduce the effect of parasitics. However, when few number of unused couplers are available this method is no longer effective. An important conclusion that we can draw from the parasitic analysis is the trade-off between the flexibility in control and the mesh loss. In fact, by increasing the mesh size we will have lower utilization of the mesh and better control of parasitics. On the other side, smaller size meshes offer better loss performance but we loose our control over parasitics.

7.2 DACs Discretization Effect

We also studied different actuation schemes for the MZI-based couplers which can be incorporated as optical gates in the FP-PICs. This study was done for both MEMS-based and heater-based couplers. The main goal of this study was to show that by using proper scheme (co-tuning of the phase shifters on both arms) we can minimize the effect of discretization errors caused by the DACs as the drivers. The importance of this investigation was that it demonstrated that lower resolution DACs can be used as an alternative to high-resolution ones using proposed architecture.

In large-scale circuits, where hundreds or thousands of such electronic components are required, utilizing low-resolution DACs can reduce costs and enhance the speed of mesh configuration. Low-resolution DACs are generally faster because they have fewer bits to process, shorter settling times, and simpler internal architectures. High-resolution DACs, while providing greater precision, often operate at slower speeds due to their increased complexity and need for higher accuracy. As an example for the cost comparison, the price of an 8-bit DAC can range from a few cents to a few dollars, while a 16-bit DAC can range from a couple of dollars to tens of dollars, depending on performance, accuracy, and other features. However, we should note that the use of two lower-precision DACs might imply more complicated control, as two DACs need to be operated together.

7.3 FP-PIC Circuit Design

Chapter 3 started with presenting the mesh analysis performed to determine the final mesh architecture of the FP-PICs for the MORPHIC RUN2 and RUN3; and, it ended with detailing the circuit designs made for the programmable circuits. To design a recirculating mesh we needed to decide the topology, shape, and size of the mesh. For the topology, we chose the hexagonal cells as suggested by Daniel Perez [30], because this topology outperforms the triangle and square topology in terms of flexibility, switching elements per area, and reconfiguration performance. To determine the shape and size of the meshes we performed both scaling and loss analysis. Based on our analysis, rectangular-shaped meshes are more promising candidates for FP-PICS compared to the other mesh shapes like radial-shaped meshes.

In MORPHIC RUN2, we decided to design a 7-cell hexagonal mesh for both MEMS-based and heater-based circuits; however, the FP-PIC circuit was redesigned to have a 24-cell mesh (we could receive more compact components) with a customized shape dictated by the available space on the chip. For MORPHIC RUN3, we designed a 126-cell FP-PIC based on a 9×14 parallelogram mesh. This circuit can accommodate a 16×16 is connected to a specialized unit which is a 1×16 phased array antenna.

7.4 System Architecture

The system architecture and integration for the fabricated programmable circuits are discussed in Chapter 4. One of the important steps in the packaging is the layout design of the interposers and PCB interconnects; multilayer interposers and PCB interconnects were designed by the external companies and I designed the single-layer interposers and PCB interconnects. Considering the fact that we had various circuits with different sizes on the chips, we implemented a modular control hardware where Tyndall designed the EIC boards and low-level APIs for electronic hardware. In general, to implement an electro-optic system for large-scale FP-PICs we require a multilayer technology stack starting from physical components on the photonic chips up to the top layers where the high level functionalities and user interaction/experience should be considered.

Design trade-offs are one of the key aspects of the system architecture design and implementation. In fact, the integration of MEMS components in chip design involves several trade-offs. Footprint allocation is challenged by the space demands of bond-pads, fiber couplers, and alignment markers, as well as wide rims around

air-clad MEMS cavities and silicon lids, which constrain circuit placement. Circuit construction can either involve individual MEMS blocks or integrated subcircuits; while the latter allows for greater complexity, routing limitations remain. Pattern density is constrained by the inability to use dummies in MEMS regions, necessitating careful design for fabrication viability. Lastly, grounding considerations impact electrical isolation and design flexibility. While isolating MEMS components reduces crosstalk but causes signal loss, maintaining waveguides at ground potential—chosen here—ensures minimal crosstalk, albeit at the cost of geometric flexibility, with positive performance results confirmed in measurements.

7.5 Software Framework

In Chapter 5, we elaborated the software framework, Borna, developed for generic field-programmable photonic integrated circuits (FP-PICs). Various layers and abstractions used in the framework were discussed and example codes for the different situations were demonstrated. The main goal of the framework is to facilitate the design process, circuit simulation, and mesh configuration of the programmable circuits. It abstracts the complexity of low-level hardware operations, enabling developers to program FP-PICs without needing in-depth knowledge of the physical photonic components. This allows users to focus on designing functionality rather than managing intricate hardware details. The framework is built using Python and utilizes external libraries for graph-based routing (Graphspay), circuit simulation (IPKISS), and low-level API drivers (developed by Tyndall). As a showcase of Borna, we used it to configure the NOVA chip on which a 7-cell heater-based FP-PIC has been fabricated. This demonstration shows that the flexibility of the framework to be used for other hardware and programmable circuits.

7.6 Chip Characterizations

Building complex circuits with suspending $220\,\mathrm{nm}$ thin photonic device layers is a challenging task that requires tremendous teamwork, debugging, and several design and fabrication iterations. As shown in Chapter 6, all the fabricated MEMS-circuits have failed and we could not observe light transmission from the input to the outputs. Therefore, to find out the reason behind such a low yield, physical yield analysis, inspecting the circuits using multiple means such as backreflection measurements, destructive tests and IR camera inspection through the sealing lids to quantify the actual yield was performed by the consortium. For backreflection measurement we diagnosed various switch circuits (Crossbar 4×4, PI-loss 4×4,

Benes 4×4, and Benes 16×16) on both non-packaged and packaged chips. Our measurements confirm that the fabricated MEMS blocks are not as robust as what we expected. Also, using backreflection measurement and analysis we could inspect the internal blocks of the circuit and observe their non-functionality. I believe it is an informative approach to analysis internal behavior of the large scale circuits.

Despite of the characterization techniques carried out during the MORPHIC project, we are not still able to explain the reason of the MEMS collapse in the circuits and further investigation is required. For the future designs, we should take greater care to include debugging features and test sites that can be interrogated during the packaging process.

7.7 Future Improvements

An important next step for the future works is to identify the root cause of the MEMS collapse in the circuits and develop methods to prevent it from occurring. It is worth mentioning that such investigation is being further pursued in the PHORMIC project. Based on my experience in the circuits characterization for both packaged and non-packaged chips (etched and sealed) and as it was shown for the actuation test for the full demonstrator and the test node, we should perform optical transmission measurement of the circuits in the following steps:

- After releasing the MEMS devices on the etched chips (before hermetic sealing)
- After completing the hermetic sealing process (sealed chips).
- Right after the shipment of the sealed chips before starting the packaging process.
- After completing each step of the packaging process.

I should mention that for these measurements, we should first perform the passive measurement for all the circuits on the chips and then start actuating the components one by one and then configure the circuits by co-actuating of several couplers and phase shifters.

On the other side, we could revise the MEMS components implementation including the actuators and also the substrate grounders. For example, we can incorporate stiffer MEMS or provide more reliable potential for substrate grounders by modifying the back-side layers of the chip. Additionally, we can implement

ESD protection diodes into the iSiPP50G platform (for the bondpads grid), and use methods to avoid ultrasonic vibrations during wirebonding and stud bumping.

Assuming the MEMS circuits have a high yield and function as intended, what further improvements or advancements could we pursue? One limiting factor in scaling of the programmable circuit is loss, obviously by increasing the size of the meshes, optical losses become more pronounced. To compensate the mesh loss in very large-scale FP-PICs, gain materials can help us. To do this we can either implement them on the side of the mesh as we did for the 126-cell FP-PIC in MORPHIC RUN3, or carefully design a non-uniform mesh including islands containing amplifiers.

In MORPHIC, we selected flexibility over the compactness. Instead of custom electronic driver ASIC, an approach based on discrete electronics arranged in a flexible and extensible architecture was chosen. Hence, the consortium implemented a modular design using electronic interface and control (EIC) boards because of unknown number of devices the electrical characteristics. One of the solutions to improve performance of EICs is to add FPGAs onto EIC boards to speed up the SPI lines which limits the update frequency of the circuits. In fact, the FPGAs can control many ICs in parallel with refresh rate of few milliseconds.

As described in Chapter 4, the MEMS devices have been placed under vacuum and afterwards hermetically sealed using silicon. However, it may be possible to use other materials such as silica or silicon carbide to allow visual inspection without the need for IR cameras. Co-integration of the silicon and silicon nitride also can improve the robustness of the MEMS components; however, we should note that silicon nitride is not a conductive material and we will need conducting layers as well. Other platforms such as VTT offer thicker silicon layer compared to IMEC; however, they have thinner oxide layer. Hence, further investigation is required for such platforms.

7.8 Importance of the MEMS

I should remind and emphasize that MEMS devices consume very little power (nW/device) compared to the heaters ($>10\,\mathrm{mW}$/device) and unlike them MEMS actuators can be positioned close to each other without any concern about thermal cross talk. In quantum applications, we are operating in a cryogenic environment, hence MEMS are the better candidate. Additionally, they have similar insertion loss and optical path lengths, and can operate 10 times faster [110, 111]. This gives us a great opportunity to build large-scale programmable circuits with higher efficiency and performance. As result, resolving the current change of MEMS collapse in the

circuit will open a new avenue for fantastic scientific and engineering applications.

A

Measurement Coding Examples

In this appendix, we present the python scripts for utilizing active and passive measurement units embedded in *Borna* framework.

A.1 Components Import

```
import numpy as np
from borna.general import load_yaml_file
# hardware
from borna.hardware.polatis.Polatis_switch import Polatis
from borna.hardware.luna.Luna_tcpip.Luna_OVA import Luna_OVA
from borna.hardware.eic_boards.eic_boards import EIC
from pymeasure.instruments.Keithley.voltage_current_source
    import Keithley2400
from pymeasure.units.unit import AMPERE, MILLIAMPERE,
    MICROAMPERE, VOLT

#
from borna.circuits.CROSSBAR4x4.pics import RUN2CrossBar4x4

# measurements
from borna.measurements import AMU001
```

A.2 Netlists and Calibration Parameters

```

# NETLISTS/CALIBRATION PARAMS
# -------------------------
# 1) Optical Switch Connections
switch_map = {'in1': 2, 'in2': 3, 'in3': 4, 'in4': 5,
              'out1': 6, 'out2': 7, 'out3': 8, 'out4': 9,
              'luna_source': 56, 'luna_read': 57}

# 2) The netlist of blocks id on the schematic and their corresponding
# DAC/ADC pins extracted from the master netlist
netlist_path = 'netlists/pic_netlist.yaml'
pic_netlist = load_yaml_file(netlist_path)

# 3) Calibration Data
eic_pd_data_path = 'netlists/pd_cal_1d.yaml'
pd_cal_1d = load_yaml_file(eic_pd_data_path)
```

A.3 Device Initialization

```

# INIT DEVICES
# ----------------
# Optical Source: LUNA
Luna = Luna_OVA(name="luna_ova", host='localhost',
                port=9, timeout=2, connection_attempts=5)
Luna.connect()

# Optical Switch: Polatis
o_switch = Polatis()
o_switch.map = switch_map

# Voltage Source: Keithley
K = Keithley2400(name='Keithley SMU', address="GPIB0::8::INSTR")
K.mode = 'voltage'
K.source_voltage_range = 11 * VOLT
K.current_limit = 105.0 * MICROAMPERE

# EIC boards
eic1 = EIC(board_model='EIC0001D',
           board_nr='bb1_eic1',
           ip_address='192.168.7.2',
           port_nr='6660',
```

```
            pd_cal=pd_cal_1d)

# Photonic Circuit
pic = RUN2CrossBar4x4(eic=eic1)
pic.netlist = pic_netlist
```

A.4 Measurement Scenarios

Here we demonstrate three scenarios for active measurements and an example for performing a passive measurement. Although these are simple example, they clearly show the flow for implementing control loops.

A.4.1 single coupler tunning and full optical ports sweep

In this example, we change coupling values of the *cp_11* of the crossbar4x4 switch circuit, and sweep all the optical ports pair for transmission measurements.

```
# init measurement unit
m_unit = AMU001(luna=Luna, o_switch=o_switch, pic=pic,
                dir_path='results/measurement_001')

# define tasks
task_list = []
for coupling in np.linspace(0, 1, num=20):
    temp_task = {
        'config': {'cps': {(1, 1): coupling}} ,
        'ports_pairs': pic.ports_pairs_combination.all,
        'pds_list': None}
    task_list.append(temp_task)

m_unit.update_task_list(task_list)

# execute measurement
m_unit.run()
```

A.4.2 single coupler tunning based on optical transmission response

In this example, we first set the coupling of *cp_11* ($\kappa = 1$). Then optical transmission of the ('in1', 'out1') is measured. Next, we apply a simple control algorithm:

if the transmission at 1.55 μm is greater than -18.0dB, the coupling value should be set as $\kappa = 0.5$; otherwise, it should be 1.0.

```

# init measurement unit
m_unit = AMU001(luna=Luna, o_switch=o_switch, pic=pic,
                dir_path='results/measurement_002')

# define tasks
cp_coords_1 = (1, 1)
task_1 = {
    'config': {'cps': {cp_coords_1: 0.0}} ,
    'ports_pairs': [('in1', 'out1')],
    'pds_list': None}
task_list = [task_1]
m_unit.update_task_list(task_list)

# execute measurement
m_unit.run()

wavelength_index = m_unit.get_wavelength_index(1.55)  # um
o_power_1 = -18.0 # dB

if m_unit.o_transmission[('in1', 'out1')][wavelength_index]
    > o_power_1:  # dB
    task_1['config']['cps']['cp_coords_1']= 0.5
    task_list = [task_1]
    m_unit.update_task_list(task_list)
    m_unit.run()
else:
    task_1['config']['cps']['cp_coords_1']= 1.0
    task_list = [task_1]
    m_unit.update_task_list(task_list)
    m_unit.run()
```

A.4.3 single coupler tunning based on PD read power and optical transmission response

This example is similar to the previous example demonstrate a simple control algorithm with the addition of photodiods power constraint.

```
# init measurement unit
m_unit = AMU001(luna=Luna, o_switch=o_switch, pic=pic,
        dir_path='results/measurement_003')

# define tasks
cp_coords_1 = (3, 4, 1)
```

```
task_1 = {
    'config': {'cps': {cp_coords_1: 0.0}} ,
    'ports_pairs': [('p1', 'p3')],
    'pds_list': ['pd1']}
task_list = [task_1]
m_unit.update_task_list(task_list)

#execute measurement
m_unit.run()

wavelength_index = m_unit.get_wavelength_index(1.55)  # um
o_power_1 = -18.0 # dB
pd_power_1 = 0.1 # mWatt

if (m_unit.o_transmission[('p1', 'p3')][wavelength_index] >
    o_power_1) & (m_unit.pd_powers['pd1'] > pd_power_1):
    task_1['config']['cps']['cp_coords_1']= 0.5
    task_list = [task_1]
    m_unit.update_task_list(task_list)
    m_unit.run()
else:
    task_1['config']['cps']['cp_coords_1']= 1.0
    task_list = [task_1]
    m_unit.update_task_list(task_list)
    m_unit.run()
```

A.4.4 Passive Measurement

This example shows that how easy users can perform a passive measurement (no actuation is applied) by selecting desire optical ports and photodiods.

```
# initialize measurement unit
m_unit = PMU001(luna=Luna, o_switch=o_switch, pic=pic,
dir_path='results/measurement_004')

# define photodiods to be read
m_unit.pds_list = ['pd1', 'pd2']

# define optical ports for transmission measurements
m_unit.ports_pairs = [('p1', 'p2'), ('p1', 'p3')]

# execute measurement
m_unit.run()
```

B

Summary of my Contributions in MORPHIC

This appendix aims to summarize and clarify my contributions to the MORPHIC project. During the project, I have been involved in various activities including photonic circuit analysis and design, overall system architecture, hardware design for the photonic chip's control system, software development for the control system, and photonic measurements on the fabricated devices.

B.1 PIC Design and Characterization

Table B.1 provides a summary of my contributions to the design and characterization of photonic chips, as well as the design of their corresponding hardware interfaces, including interposers and PCB interconnects. The MEMS couplers and phase shifters (Sec. 3.2.3) were designed by H. Sattari (EPFL) and P. Edinger (KTH). Switch circuits (Sec. 3.2.7) were designed by B. Abasahl (IMEC) and I was responsible to design the FP-PIC circuits and the test nodes. And, 7-cell FP-PIC on NOVA chip (Sec. 5.12) was designed by my colleagues L. Van Iseghem and X. Chen. The summary of the demonstrator circuits on MORPHIC RUN2 and RUN3 as well as 7-cell FP-PIC circuit on the NOvA chip are shown in Fig. B.1.

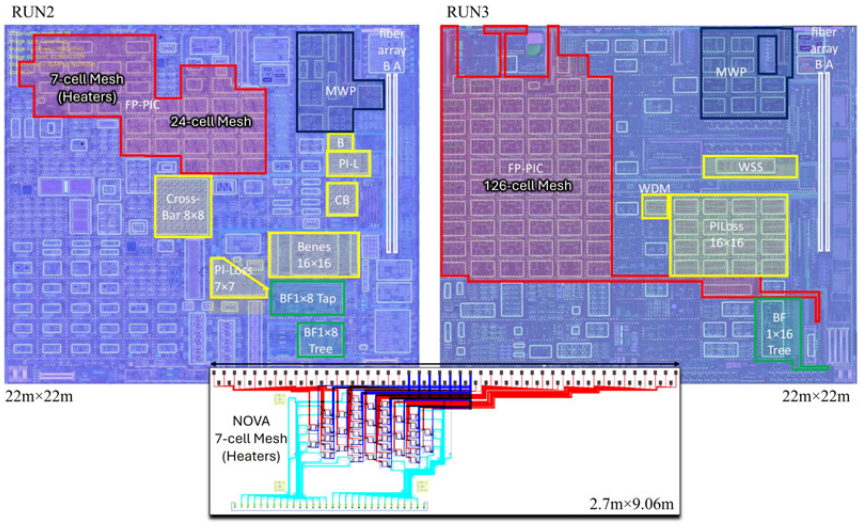


Figure B.1: Demonstrator circuits on MORPHIC RUN2 (left) and RUN3 (right). The different circuits relate to switching (yellow), beamforming (green), microwave photonics (navy) and the FP-PIC (red). Each of these circuits is connected either to Fiber Array A or B. And, the lower image shows the layout of the heater-based 7-cell FP-PIC on the Nova chip.

As it has been discussed in Chapter 4, Tyndall implemented two types of packaging: Mini Demonstrators (two versions: MD1 and MD2) and Full Demonstrators (FD). Both types of the demonstrators include switch circuits as listed in Table B.1. MEMS-based 24-cell (Sec. 3.2.4) and 126-cell (Sec. 3.2.5) FP-PIC circuit were included in the mini and full demonstrators, respectively. However, we could not receive their packaged chips during the project. I designed the interposers and PCB interconnects for the mini demonstrators, while those for the large-scale full demonstrators were developed by the Tyndall external partners.

For the heater-based 7-cell FP-PIC on RUN2 (Sec. 3.2.2), I designed both the interposer and PCB interconnect. Nevertheless, this circuit was not packaged due to the MORPHIC timeline. Fortunately, another heater-based 7-cell FP-PIC became available during the writing of this thesis, allowing me to demonstrate the Borna framework for its control and configuration (Sec. 5.12). This circuit was designed by my colleagues Xiang Chen and Lukas Van Iseghem.

For the non-packaged chips, I measured FP-PIC and switch circuits across multiple samples for three different chip types: unprocessed, etched, and sealed (see Sec. 6.1 for more details).

Circuits	RUN	PIC Design	Interposer Design	PCB Interconnect Design	Netlist	Characterization	
						Non-packaged	Packaged
Crossbar 4×4	2	-	✔	✔	✔	U-E-S	MD1-MD2-FD
PI loss 4×4	2	-	✔	✔	✔	U-E-S	MD2-FD
Benes 4×4	2	-	✔	✔	✔	U-E-S	MD2-FD
Benes 16×16	2	-	✔	✔	✔	U-E-S	MD1-MD2-FD
7-cell FP-PIC *	2	✔	✔	✔	✔	U	not available
24-cell FP-PIC	2	✔	✔	✔	✔	U-E-S	not available
126-cell FP-PIC	3	✔	-	-	✔	U-E-S	not available
Test Nodes	3	✔	-	-	-	U-E-S	not available
7-cell FP-PIC ** (NOVA)	-	-	-	-	-	-	MD

*Table B.1: Summary of my collaborations. Non-packaged chips consist of unprocessed (U), etched (E), and sealed (S) versions as explained in Chapter 6. MD and FD refer to mini demonstrators and full demonstrators, respectively. * 7-cell FP-PIC circuit is based on heaters. ** Another heater-based FP-PIC circuit fabricated on the NOVA chip.*

B.2 Mesh Analysis and Modeling

As detailed in Chapters 2 and 3, I conducted extensive mesh analysis, including parasitics, loss, and scaling evaluations. This involved performing circuit simulations, exploring various mesh configurations, analyzing circuit components with different loss characteristics, and implementing graph-based algorithms. The Python code developed for these computations was subsequently integrated into the software framework.

B.3 Software Framework

One of my primary contributions to the MORPHIC project was the development of the python-based software framework detailed in Chapter 5. I collaborated closely with colleagues at UGent and Tyndall to build this framework. The low-level module for electronic control was developed by Dr. C. Antony at Tyndall, while the graph-based routing algorithm library was created by X. Chen at UGent. For more clarity, Fig. B.2 highlights contributions from other collaborators to the framework.

I should remind that the software framework was used to characterize the demonstrators. Additionally, I used the framework to control the 7-cell FP-PIC on the NOVA chip, demonstrating its flexibility to support other drivers and FP-PICs.

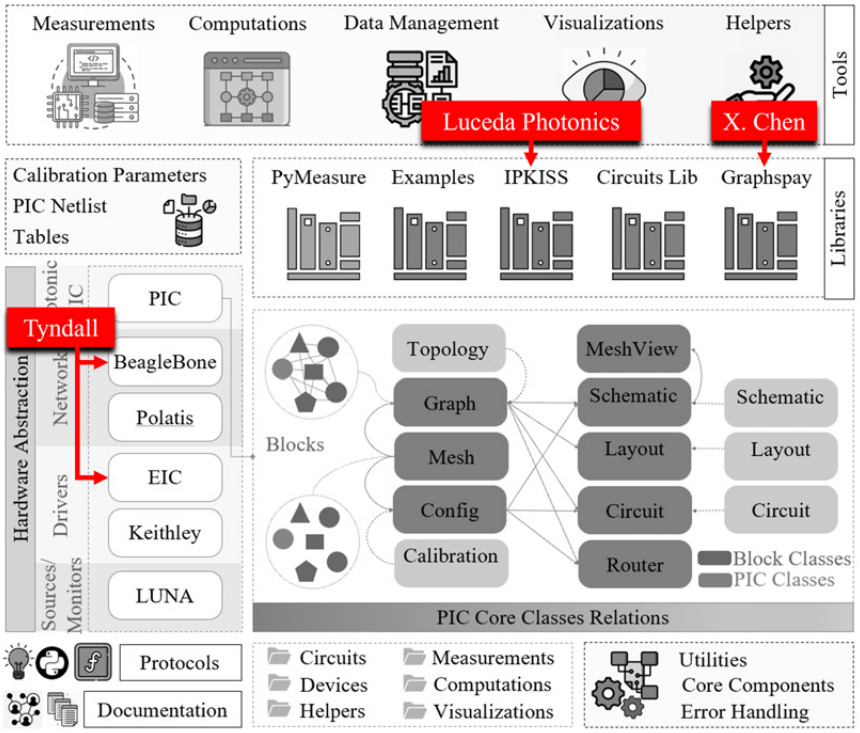


Figure B.2: An overview of the Borna architecture, as discussed in Chapter 5, highlighting contributions from other collaborators to the framework.

B.4 Netlists

Another key responsibility was to generate various netlists (Table B.1) based on the mapping of all hardware connections, including DAC/ADC pins, interposer and PCB interconnect pads, DC pads on the PICs, and the abstract circuit mesh components displayed in the GUI layer (Sec. 5.8). To support this process, I had a close communication with the packaging group in Tyndall, wrote python scripts for cross-checking, and utilized Altium software.

B.5 EIC Boards

For the control electronics, my role involved testing and characterizing the EIC boards received from Tyndall. I also provided feedback to refine and redesign the EIC board architecture. For instance, the design of the readout circuits was based on

the specifications and architecture of the balanced photodiodes (BPDs) and monitor PDs integrated into the photonic circuits.

Additionally, I analyzed three different architectures (Fig.B.3) for MZI-based couplers to address the impact of DAC discretization on the coupling response of these circuit blocks. This analysis was conducted for both MEMS-based and heater-based MZI couplers. As detailed in Sec.**??**, the results indicate that employing an MZI with a $\pi/2$ phase delay (quadrature) and heaters on both arms, where phase shifters are used in combination, creates a discrete 2D coupling space. This approach enhances coupling resolution, allowing the use of lower-resolution DACs instead of higher-resolution ones.

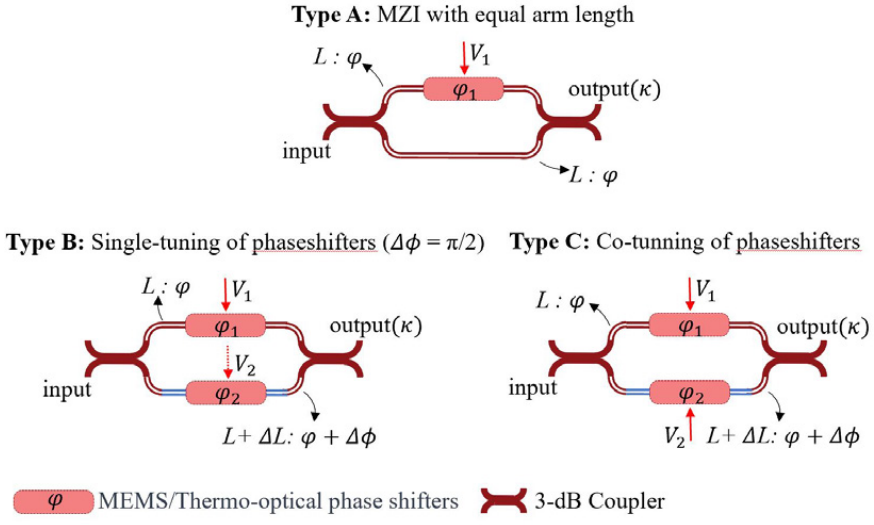


*Figure B.3: Tunable 2 × 2 MZI couplers phase shifters. **Type A**: MZI with equal arm lengths is loaded with an actuator on one arm. **Type B**: MZI with a $\pi/2$ phase delay (quadrature) loaded with actuators on both arms. In type B, the coupler is operated in push-pull, where only one of the phase shifters operating at any given time. **Type C**: MZI with a $\pi/2$ phase delay (quadrature) loaded with heaters on both arms. In type C, the phase shifters are used together creating discrete 2D-space for the coupling.*

References

[1] S. Shekhar, W. Bogaerts, L. Chrostowski, J. E. Bowers, M. Hochberg, R. Soref, and B. J. Shastri, "Roadmapping the next generation of silicon photonics," *Nature Communications*, vol. 15, p. 751, 2024.

[2] D. Pérez-López, A. M. Gutierrez, E. Sánchez, P. DasMahapatra, and J. Capmany, "Integrated photonic tunable basic units using dual-drive directional couplers," *Optics Express*, vol. 27, p. 38071, 2019.

[3] N. C. Harris, G. R. Steinbrecher, M. Prabhu, Y. Lahini, J. Mower, M. Mondol, T. Baehr-Jones, M. Hochberg, S. Lloyd, and D. Englund, "Quantum transport simulations in a programmable nanophotonic processor," *Nature Photonics*, vol. 11, pp. 447–452, 2017.

[4] W. Bogaerts, "Programmable photonics," *Optics and Photonics News (invited)*, vol. 35, no. 3, pp. 34–41, 2024.

[5] M. Pantouvaki *et al.*, "Active components for 50 gb/s nrz-ook optical interconnects in a silicon photonics platform," *Journal of Lightwave Technology*, vol. 35, pp. 631–638, 2017.

[6] C. Errando-Herranz, A. Y. Takabayashi, P. Edinger, H. Sattari, K. B. Gylfason, and N. Quack, "Mems for photonic integrated circuits," *IEEE Journal of Selected Topics in Quantum Electronics*, vol. 26, no. 2, pp. 1–16, 2020.

[7] A. Y. Takabayashi, *Microelectromechanical Systems (MEMS) for Silicon Photonics*. PhD thesis, École Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland, November 2021. Ph.D. thesis, Thesis number 9413.

[8] J. Zhang, M. Muneeb, S. Stankovic, S. Latkowski, A. Abbasi, G. Roelkens, and R. Baets, "Transfer-printing-based integration of a iii–v-on-silicon distributed feedback laser," *Optics Express*, vol. 26, pp. 8821–8830, 2018.

[9] J. Zhang, L. Bogaert, C. J. Krückel, E. Soltanian, H. Deng, B. Haq, J. Rimbock, J. V. Kerrebrouck, G. Lepage, P. Verheyen, J. V. Campenhout, P. Ossieur, D. V. Thourhout, G. Morthier, W. Bogaerts, and G. Roelkens,

"Micro-transfer printing inp c-band soas on advanced silicon photonics platform for lossless mzi switch fabrics and high-speed integrated transmitters," *Optics Express*, vol. 31, no. 26, pp. 42807–42821, 2023.

[10] P. Edinger, A. Y. Takabayashi, C. Errando-Herranz, U. Khan, H. Sattari, P. Verheyen, W. Bogaerts, N. Quack, and K. B. Gylfason, "Silicon photonic microelectromechanical phase shifters for scalable programmable photonics," *Optics Letters*, vol. 46, p. 5671, November 2021.

[11] N. C. Harris, D. Bunandar, M. Pant, J. C. Steinbrecher, S. Mower, M. Prabhu, T. Baehr-Jones, M. Hochberg, X. Sun, J. B. Larson, N. Carolan, F. Marsili, M. D. Shaw, S. Lloyd, J. L. O'Brien, D. Englund, and M. Soljačić, "Linear programmable nanophotonic processors," *Optica*, vol. 5, pp. 1623–1631, 2018.

[12] D. A. B. Miller, "Self-configuring universal linear optical component," *Photonics Research*, vol. 1, pp. 1–15, 2013.

[13] W. R. Clements, P. C. Humphreys, B. J. Metcalf, W. S. Kolthammer, and I. A. Walmsley, "An optimal design for universal multiport interferometers," *Optica*, vol. 3, pp. 1460–1465, 2016.

[14] A. Ribeiro, A. Ruocco, L. Vanacker, and W. Bogaerts, "Demonstration of a 4×4-port universal linear circuit," *Optica*, vol. 3, pp. 1348–1357, 2016.

[15] S. Pai, Y. Huang, H. Wang, X. Meng, N. M. L. Athanasopoulos, R. F. Ferreira, D. Pérez-López, J. Capmany, A. E. J. Lim, and K. Bergman, "Parallel programming of an arbitrary feedforward photonic network," *IEEE Journal of Selected Topics in Quantum Electronics*, vol. 25, p. 6100813, 2020.

[16] M. Reck, A. Zeilinger, H. J. Bernstein, and P. Bertani, "Experimental realization of any discrete unitary operator," *Physical Review Letters*, vol. 73, pp. 58–61, 1994.

[17] J. Capmany and D. Perez, *Programmable Integrated Photonics*. Oxford University Press, 2020.

[18] L. Zhuang, C. G. H. Roeloffzen, M. Hoekman, K. Boller, and A. J. Lowery, "Programmable photonic signal processor chip for radiofrequency applications," *Optica*, vol. 2, pp. 854–859, 2015.

[19] D. Perez, I. Gasulla, J. Capmany, W. A. Cortes, and L. F. Beltran, "Silicon photonics rectangular universal interferometer," *Laser & Photonics Reviews*, vol. 11, p. 1700219, 2017.

[20] L. Lu, L. Zhou, and J. Chen, "Programmable scow mesh silicon photonic processor for linear unitary operator," *Micromachines*, vol. 10, p. 646, 2019.

[21] Y. Shen, N. C. Harris, S. Skirlo, M. Prabhu, T. Baehr-Jones, X. Sun, S. Zhao, H. Larochelle, D. Englund, and M. Soljačić, "Deep learning with coherent nanophotonic circuits," *Nature Photonics*, vol. 11, pp. 441–446, 2017.

[22] D. Perez, I. Gasulla, and J. Capmany, "Programmable multifunctional integrated nanophotonics - review article," *De Gruyter - Nanophotonics*, vol. 7, pp. 1351–1371, 2018.

[23] N. Quack, A. Y. Takabayashi, H. Sattari, P. Edinger, G. Jo, S. J. Bleiker, C. Errando-Herranz, K. B. Gylfason, F. Niklaus, U. Khan, P. Verheyen, A. K. Mallik, J. S. Lee, M. Jezzini, I. Zand, P. Morrissey, C. Antony, P. O'Brien, and W. Bogaerts, "Integrated silicon photonic mems," *MicroSystems & Nanoengineering*, vol. 9, p. 27, 2023.

[24] D. A. B. Miller, "Perfect optics with imperfect components," *Optica*, vol. 2, no. 8, pp. 747–750, 2015.

[25] F. Morichetti, S. Grillanda, M. Carminati, G. Ferrari, M. Sampietro, M. J. Strain, M. Sorel, and A. Melloni, "Non-invasive on-chip light observation by contactless waveguide conductivity monitoring," *IEEE Journal of Selected Topics in Quantum Electronics*, vol. 20, pp. 292–301, 2014.

[26] H. Jayatilleka, H. Shoman, L. Chrostowski, and S. Shekhar, "Photoconductive heaters enable control of large-scale silicon photonic ring resonator circuits," *Optica*, vol. 6, pp. 84–91, 2019.

[27] S. K. Selvaraja, P. D. Heyn, G. Winroth, P. Ong, G. Lepage, C. Cailler, A. Rigny, K. K. Bourdelle, W. Bogaerts, D. V. Thourhout, J. V. Campenhout, and P. Absil, "Highly uniform and low-loss passive silicon photonics devices using a 300mm cmos platform," in *Optical Fiber Communication Conference (OFC)*, p. Th2A.33, Optical Society of America, March 2014.

[28] W. Bogaerts, D. Taillaert, B. Luyssaert, P. Dumon, J. V. Campenhout, P. Bienstman, D. V. Thourhout, R. Baets, V. Wiaux, and S. Beckx, "Basic structures for photonic integrated circuits in silicon-on-insulator," *Optics Express*, vol. 12, p. 1583, April 2004.

[29] W. Bogaerts and S. K. Selvaraja, "Compact single-mode silicon hybrid rib/strip waveguide with adiabatic bends," *IEEE Photonics Journal*, vol. 3, pp. 422–432, June 2011.

[30] D. Pérez and J. Capmany, "Scalable analysis for arbitrary photonic integrated waveguide meshes," *Optica*, vol. 6, pp. 19–27, 2019.

[31] X. Chen, M. M. Milosevic, S. Stankovic, S. Reynolds, T. D. Bucio, K. Li, and G. T. Reed, "The emergence of silicon photonics as a flexible technology platform," *Proceedings of the IEEE*, vol. 106, pp. 2101–2116, 2018.

[32] M. Smit, K. Williams, and J. van der Tol, "Past, present, and future of inp-based photonic integration," *APL Photonics*, vol. 4, p. 050901, 2019.

[33] A. Rahim, T. Spuesens, R. Baets, and W. Bogaerts, "Open-access silicon photonics: current status and emerging initiatives," *Proceedings of the IEEE*, vol. 106, pp. 2313–2330, 2018.

[34] P. Muñoz, P. W. L. van Dijk, D. Geuzebroek, M. Geiselmann, C. Domínguez, A. Stassen, J. D. Doménech, M. Zervas, A. Leinse, C. G. H. Roeloffzen, B. Gargallo, R. Baños, J. Fernández, G. Mico-Cabanes, L. A. Bru-Orgiles, and D. P. Abellán, "Foundry developments toward silicon nitride photonics from visible to the mid-infrared," *IEEE Journal of Selected Topics in Quantum Electronics*, vol. 25, no. 5, p. 8200513, 2019.

[35] R. Soref, "The past, present, and future of silicon photonics," *IEEE Journal of Selected Topics in Quantum Electronics*, vol. 12, pp. 1678–1687, November 2006.

[36] C. K. et al., "Silicon photonic circuits: On-cmos integration, fiber optical coupling, and packaging," *IEEE Journal of Selected Topics in Quantum Electronics*, vol. 17, pp. 498–509, May 2011.

[37] A. Rickman, "The commercialization of silicon photonics," *Nature Photonics*, vol. 8, no. 8, pp. 579–582, 2014.

[38] P. D. et al., "Submilliwatt, ultrafast and broadband electro-optic silicon switches," *Optics Express*, vol. 18, no. 24, pp. 25225–25231, 2010.

[39] A. L. Washburn, L. C. Gunn, and R. C. Bailey, "Label-free quantitation of a cancer biomarker in complex media using silicon photonic microring resonators," *Analytical Chemistry*, vol. 81, pp. 9499–9506, November 2009.

[40] L. C. et al., "Silicon photonic resonator sensors and devices," p. 823620, 2012.

[41] L. R. Chen, "Silicon photonics for microwave photonics applications," *Journal of Lightwave Technology*, vol. 35, pp. 824–835, February 2017.

[42] W. Bogaerts, A. Rahim, B. Snyder, B. Maes, P. Dumon, J. D. Doménech, R. Baños, J. Fernández, M. Lamponi, A. Higuera-Rodríguez, A. E. Lim, G. Li, T. Helin, X. Wang, S. Li, and J. V. Campenhout, "Programmable photonics: An opportunity for an accessible large-volume pic ecosystem,"

IEEE Journal of Selected Topics in Quantum Electronics, vol. 26, no. 5, p. 8302517, 2020.

[43] D. Marpaung, J. Yao, and J. Capmany, "Integrated microwave photonics," *Nature Photonics*, vol. 13, pp. 80–90, 2019.

[44] J. Carolan, C. Harrold, C. Sparrow, E. Martín-López, N. J. Russell, J. W. Silverstone, P. Shadbolt, N. Matsuda, M. Oguma, M. Itoh, G. D. Marshall, M. G. Thompson, J. C. F. Matthews, T. Hashimoto, J. L. O'Brien, and A. Laing, "Universal linear optics," *Science*, vol. 349, pp. 711–716, 2015.

[45] N. C. Harris, Y. Sung, J. Mower, Y. Zhang, C. Steinbrecher, M. Smith, T. Baehr-Jones, M. Hochberg, X. Li, S. Assefa, F. Marsili, J. B. Chang, C. L. Schuck, M. D. Shaw, K. Berggren, V. Anant, M. Soljačić, and D. Englund, "Large-scale quantum photonic circuits in silicon," *Nanophotonics*, vol. 5, pp. 456–468, 2016.

[46] J. Notaros, C. W. Holzwarth, M. S. Anderson, A. L. Lentine, S. D. Dyer, A. L. Starbuck, D. C. Trotter, R. P. Mirin, R. M. Camacho, M. L. Merrill, A. Baumbauer, J. M. Shainline, S. W. Nam, C. T. Santori, and S. J. B. Yoo, "Programmable dispersion on a photonic integrated circuit for classical and quantum applications," *Optics Express*, vol. 25, pp. 21275–21285, 2017.

[47] F. Gan, T. Barwicz, M. Popovic, M. Dahlem, C. Holzwarth, P. Rakich, H. Smith, E. Ippen, and F. Kartner, "Maximizing the thermo-optic tuning range of silicon photonic structures," in *2007 Photonics in Switching*, pp. 67–68, IEEE, 2007.

[48] A. Masood, M. Pantouvaki, D. Goossens, G. Lepage, P. Verheyen, D. V. Thourhout, P. Absil, and W. Bogaerts, "Cmos-compatible tungsten heaters for silicon photonic waveguides," in *IEEE International Conference on Group IV Photonics GFP*, pp. 234–236, 2012.

[49] Q. Fang, J. F. Song, T.-Y. Liow, H. Cai, M. B. Yu, G. Q. Lo, and D.-L. Kwong, "Ultralow power silicon photonics thermo-optic switch with suspended phase arms," *IEEE Photonics Technology Letters*, vol. 23, no. 8, pp. 525–527, 2011.

[50] A. Masood, M. Pantouvaki, G. Lepage, P. Verheyen, J. V. Campenhout, P. Absil, D. V. Thourhout, and W. Bogaerts, "Comparison of heater architectures for thermal control of silicon photonic circuits," in *Proceedings of the 10th International Conference on Group IV Photonics*, p. ThC2, 2013.

[51] M. Milanizadeh, D. Aguiar, A. Melloni, and F. Morichetti, "Canceling thermal cross-talk effects in photonic integrated circuits," *Journal of Lightwave Technology*, vol. 37, no. 4, pp. 1325–1332, 2019.

[52] P. Sun and R. M. Reano, "Submilliwatt thermo-optic switches using free-standing silicon-on-insulator strip waveguides," *Optics Express*, vol. 18, no. 8, pp. 8406–8411, 2010.

[53] H. Sun, Q. Qiao, Q. Guan, and G. Zhou, "Silicon photonic phase shifters and their applications: A review," *Micromachines*, vol. 13, p. 1509, September 2022.

[54] J. Parra, J. Hurtado, A. Griol, and P. Sanchis, "Ultra-low loss hybrid ito/si thermo-optic phase shifter with optimized power consumption," *Optics Express*, vol. 28, no. 7, pp. 9393–9404, 2020.

[55] K. Giewont, S. Hu, B. Peng, M. Rakowski, S. Rauch, J. C. Rosenberg, A. Sahin, I. Stobert, A. Stricker, K. Nummy, F. A. Anderson, J. Ayala, T. Barwicz, Y. Bian, K. K. Dezfulian, D. M. Gill, and T. Houghton, "300-mm monolithic silicon photonics foundry technology," *IEEE Journal of Selected Topics in Quantum Electronics*, vol. 25, pp. 1–11, September 2019.

[56] Y. Fang, H. Deng, X. Chen, H. Cuma, K. P. Nagarjun, F. Ferraro, G. Lepage, P. D. Heyn, and W. Bogaerts, "Comparison of thermo-optic phase shifters in imec," in *European Conference on Integrated Optics*, 2024.

[57] W. Jin, R. G. Polcawich, P. A. Morton, and J. E. Bowers, "Piezoelectrically tuned silicon nitride ring resonator," *Optics Express*, vol. 26, pp. 3174–3187, 2018.

[58] N. Hosseini *et al.*, "Stress-optic modulator in triplex platform using a piezoelectric lead zirconate titanate (pzt) thin film," *Optics Express*, vol. 23, p. 14018, 2015.

[59] W. D. Cort, J. Beeckman, T. Claes, K. Neyts, and R. Baets, "Wide tuning of silicon-on-insulator ring resonators with a liquid crystal cladding," *Optics Letters*, vol. 36, pp. 3876–3878, 2011.

[60] Y. Xing *et al.*, "Digitally controlled phase shifter using an soi slot waveguide with liquid crystal infiltration," *IEEE Photonics Technology Letters*, vol. 27, pp. 1269–1272, 2015.

[61] C. Errando-Herranz *et al.*, "Mems for photonic integrated circuits," *IEEE Journal of Selected Topics in Quantum Electronics*, vol. 26, p. 8200916, 2020.

[62] C. Ríos *et al.*, "Integrated all-photonic non-volatile multi-level memory," *Nature Photonics*, vol. 9, pp. 725–732, 2015.

[63] M. Wuttig, H. Bhaskaran, and T. Taubner, "Phase-change materials for non-volatile photonic applications," *Nature Photonics*, vol. 11, pp. 465–476, 2017.

[64] C. Hoessbacher *et al.*, "The plasmonic memristor: a latching optical switch," *Optica*, vol. 1, p. 198, 2014.

[65] D. U. Kim, Y. J. Park, D. Y. Kim, Y. Jeong, M. G. Lim, M. S. Hong, M. J. Her, Y. Rah, D. J. Choi, S. Han, and K. Yu, "Programmable photonic arrays based on microelectromechanical elements with femtowatt-level standby power consumption," *Nature Photonics*, vol. 17, no. 12, pp. 1089–1096, 2023.

[66] J. M. Ramirez, B. Shen, H. Cai, X. Liu, Y. Shi, W. Li, Z. Cheng, J. Wang, M. Hu, J. Wen, Y. Zhou, L. He, X. Wang, W. Chen, G. Wang, S. Li, H. Lu, Y. Su, H. Ye, J. Lin, M. Wang, and W. Chen, "Iii-v-on-silicon integration: from hybrid devices to heterogeneous photonic integrated circuits," *IEEE Journal of Selected Topics in Quantum Electronics*, vol. 26, p. 6100213, 2020.

[67] D. Pérez, I. Gasulla, and J. Capmany, "Toward programmable microwave photonics processors," *Journal of Lightwave Technology*, vol. 36, pp. 519–532, 2018.

[68] G. T. Reed, G. Mashanovich, F. Y. Gardes, and D. J. Thomson, "Silicon optical modulators," *Nature Photonics*, vol. 4, pp. 518–526, 2010. corrigendum 4, 660 (2010).

[69] J. S. Lee *et al.*, "Meeting the electrical, optical, and thermal design challenges of photonic-packaging," *IEEE Journal of Selected Topics in Quantum Electronics*, vol. 22, pp. 409–417, November 2016.

[70] L. Carroll *et al.*, "Photonic packaging: Transforming silicon photonic integrated circuits into photonic devices," *Applied Sciences*, vol. 6, p. 426, December 2016.

[71] A. Annoni, E. Guglielmi, M. Carminati, S. Grillanda, P. Ciccarella, G. Ferrari, M. Sorel, M. Strain, M. Sampietro, A. Melloni, and F. Morichetti, "Automated routing and control of silicon photonic switch fabrics," *IEEE Journal of Selected Topics in Quantum Electronics*, vol. 22, pp. 169–176, 2016.

[72] S. Grillanda, M. Carminati, N. Morais, G. Ferrari, M. Sampietro, M. Strain, M. Sorel, A. Melloni, and F. Morichetti, "Non-invasive monitoring and control in silicon photonics using cmos integrated electronics," *Optica*, vol. 1, p. 129, 2014.

[73] A. Annoni *et al.*, "Unscrambling light—automatically undoing strong mixing between modes," *Light: Science & Applications*, vol. 6, p. e17110, 2017.

[74] D. A. B. Miller, "Analyzing and generating multimode optical fields using self-configuring networks," *Optica*, vol. 7, pp. 794–801, 2020.

[75] P. Dumais, D. J. Goodwill, D. Celo, J. Jiang, C. Zhang, F. Zhao, X. Tu, S. Yan, J. He, and C. Zhang, "Silicon photonic switch subsystem with 900 monolithically integrated calibration photodiodes and 64-fiber package," *Journal of Lightwave Technology*, vol. 36, pp. 233–238, 2018.

[76] D. Melati, M. Carminati, S. Grillanda, G. Ferrari, F. Morichetti, M. Sampietro, and A. Melloni, "Contactless integrated photonic probe for light monitoring in indium phosphide-based devices," *IET Optoelectronics*, vol. 9, pp. 146–150, August 2015.

[77] I. Zand, C. Antony, X. Chen, and W. Bogaerts, "Software framework architecture for programmable photonic chips," in *IEEE Photonics Society Summer Topicals*, p. WB1.3, 2022.

[78] I. Zand and W. Bogaerts, "Discretization effects of digital control of thermally tunable 2x2 mzi couplers," in *IEEE Photonics Society Summer Topicals*, p. TuE2.2, 2019.

[79] I. Zand, B. Abasahl, U. Khan, and W. Bogaerts, "Controlling parasitics in linear optical processors," in *Proceedings of the 23rd Annual Symposium of the IEEE Photonics Benelux Chapter*, pp. 152–155, 2018.

[80] I. Zand, B. Abasahl, and W. Bogaerts, "Intensity spread analysis of programmable photonic circuits with parasitics," in *IEEE Photonics Society Summer Topicals*, p. TuE2.2, 2019.

[81] I. Zand and W. Bogaerts, "Effects of coupling and phase imperfections in programmable photonic hexagonal waveguide meshes," *Photonics Research*, vol. 8, pp. 211–218, 2020.

[82] X. Chen, P. Stroobant, M. Pickavet, and W. Bogaerts, "Graph representations for programmable photonic circuits," *Journal of Lightwave Technology*, 2020. https://ieeexplore.ieee.org/document/9056549.

[83] W. Bogaerts, H. Sattari, P. Edinger, Y. A. Takabayashi, I. Zand, X. Wang, A. Ribeiro, M. Jezzini, C. Errando-Herranz, G. Talli, K. Saurav, M. G. Porcel, P. Verheyen, B. Abasahl, F. Niklaus, N. Quack, K. B. Gylfason, P. O'Brien, and U. Khan, "Morphic: Programmable photonic circuits enabled by silicon photonic mems," in *Proc. SPIE*, vol. 11285, pp. 11285–1, 2020.

[84] S. Han, T. J. Seok, N. Quack, B.-W. Yoo, and M. C. Wu, "Large-scale silicon photonic switches with movable directional couplers," *Optica*, vol. 2, p. 370, April 2015.

[85] P. P. Absil, S. Pathak, J. V. Campenhout, P. Verheyen, G. Lepage, W. Bogaerts, R. Baets, J. D. Coster, D. V. Thourhout, M. Pantouvaki, G. Roelkens, and P. Dumon, "Imec isipp25g silicon photonics: a robust cmos-based photonics technology platform," in *Proc. SPIE*, vol. 9367, p. 93670V, 2015.

[86] H. Sattari, A. Y. Takabayashi, Y. Zhang, P. Verheyen, W. Bogaerts, and N. Quack, "Compact broadband suspended silicon photonic directional coupler," *Optics Letters*, vol. 45, p. 2997, June 2020.

[87] A. Y. Takabayashi, H. Sattari, P. Edinger, P. Verheyen, K. B. Gylfason, W. Bogaerts, and N. Quack, "Broadband compact single-pole double-throw silicon photonic mems switch," *Journal of Microelectromechanical Systems*, vol. 30, pp. 322–329, April 2021.

[88] G. Jo, P. Edinger, S. Bleiker, X. Wang, A. Takabayashi, H. Sattari, N. Quack, M. Jezzini, P. Verheyen, I. Zand, U. Khan, W. Bogaerts, G. Stemme, K. Gylfason, and F. Niklaus, "Wafer-level hermetically sealed silicon photonic mems," *Photonics Research*, vol. 10, p. 14, November 2021.

[89] P. Edinger, C. Errando-Herranz, and K. B. Gylfason, "Low-loss mems phase shifter for large scale reconfigurable silicon photonics," in *Proceedings of the IEEE International Conference on Micro Electro Mechanical Systems (MEMS)*, pp. 919–921, January 2019.

[90] C. Errando-Herranz, F. Niklaus, G. Stemme, and K. B. Gylfason, "Low-power microelectromechanically tunable silicon photonic ring resonator add–drop filter," *Optics Letters*, vol. 40, no. 15, pp. 3556–3559, 2015.

[91] J. Henriksson, T. J. Seok, J. Luo, K. Kwon, N. Quack, and M. C. Wu, "Digital silicon photonic mems phase-shifter," in *2018 International Conference on Optical MEMS and Nanophotonics (OMN)*, pp. 1–2, July 2018.

[92] N. Quack, H. Sattari, A. Y. Takabayashi, Y. Zhang, W. Bogaerts, P. Edinger, C. Errando-Herranz, and K. B. Gylfason, "Mems-enabled silicon photonic integrated devices and circuits," *IEEE Journal of Quantum Electronics*, vol. PP, no. 1, p. 1, 2019.

[93] A. Takagi, K. Jinguji, and M. Kawachi, "Wavelength characteristics of (2*2) optical channel-type directional couplers with symmetric or non-symmetric coupling structures," *Journal of Lightwave Technology*, vol. 10, pp. 735–746, June 1992.

[94] Y. Akihama, Y. Kanamori, and K. Hane, "Ultra-small silicon waveguide coupler switch using gap-variable mechanism," *Optics Express*, vol. 19, pp. 23658–23663, November 2011.

[95] E. Bulgan, Y. Kanamori, and K. Hane, "Submicron silicon waveguide optical switch driven by microelectromechanical actuator," *Applied Physics Letters*, vol. 92, p. 101110, March 2008.

[96] S. Pai, B. Bartlett, O. Solgaard, and D. A. B. Miller, "Matrix optimization on universal unitary photonic devices," *Physical Review Applied*, vol. 11, p. 064044, 2019.

[97] A. Li, T. V. Vaerenbergh, P. D. Heyn, P. Bienstman, and W. Bogaerts, "Backscattering in silicon microring resonators: a quantitative analysis," *Laser & Photonics Reviews*, vol. 10, pp. 420–431, 2016.

[98] W. Bogaerts, M. Fiers, M. Sivilotti, and P. Dumon, "The ipkiss photonic design framework," in *Optical Fiber Communications Conference and Exhibition (OFC)*, pp. 1–3, 2016.

[99] D. Pérez, I. Gasulla, J. Capmany, and R. A. Soref, "Reconfigurable lattice mesh designs for programmable photonic processors," *Optics Express*, vol. 24, pp. 12093–12106, 2016.

[100] W. Bogaerts, Y. Xing, and U. Khan, "Layout-aware variability analysis, yield prediction and optimization in silicon photonic circuits," *IEEE Journal on Selected Topics in Quantum Electronics*, vol. 25, no. 5, p. 6100413, 2019.

[101] A. Li and W. Bogaerts, "Using backscattering and backcoupling in silicon ring resonators as a new degree of design freedom," *Lasers & Photonics Reviews*, p. 1800244 (18 pages), 2019.

[102] A. Li and W. Bogaerts, "Backcoupling manipulation in silicon ring resonators," *Photonics Research*, vol. 6, no. 6, pp. 620–629, 2018.

[103] M. R. Watts, J. Sun, C. DeRose, D. C. Trotter, R. W. Young, and G. N. Nielson, "Adiabatic thermo-optic mach-zehnder switch," *Optics Letters*, vol. 38, pp. 733–735, 2013.

[104] D. Oliveira, M. D. Aguiar, M. Milanizadeh, E. Guglielmi, F. Zanetto, G. Ferrari, M. Sampietro, F. Morichetti, and A. Melloni, "Automatic tuning of silicon photonics microring filter array for hitless reconfigurable add–drop," *Journal of Lightwave Technology*, vol. 37, pp. 3939–3947, 2019.

[105] I. Zand, X. Chen, and W. Bogaerts, "Application-specific scaling in programmable photonic circuits," in *European Conference on Integrated Optics*, p. 10.3, 2020.

[106] Q. Cheng, M. Bahadori, X. Wu, E. El-Fiky, Y. Zhou, B. Guan, M. Takenaka, K. Preston, and K. Bergman, "Silicon photonic switch topologies and routing strategies for disaggregated data centers," *IEEE Journal of Selected Topics in Quantum Electronics*, no. c, pp. 1–1, 2019.

[107] X. Chen and W. Bogaerts, "Me2.2 - a graph-based design and programming strategy for reconfigurable photonic circuits," in *2019 IEEE Photonics Society Summer Topical Meeting Series (SUM)*, pp. 1–2, 2019.

[108] N. Quack, H. Sattari, A. Y. Takabayashi, Y. Zhang, P. Verheyen, W. Bogaerts, P. Edinger, C. Errando-Herranz, and K. B. Gylfason, "Mems-enabled silicon photonic integrated devices and circuits," *IEEE Journal of Quantum Electronics*, vol. 56, no. 1, pp. 1–10, 2020.

[109] G. Jo and P. Edinger, "Wafer-level hermetically sealed silicon photonic mems," in *Silicon Photonic MEMS Building Blocks for Low-Power Programmable Circuits*, KTH Royal Institute of Technology, 2022.

[110] R. Baghdadi, M. Gould, S. Gupta, M. Tymchenko, D. Bunandar, C. Ramey, and N. C. Harris, "Dual slot-mode noem phase shifter," *Optics Express*, vol. 29, no. 12, pp. 19113–19119, 2021.

[111] S. Han, J. Beguelin, L. Ochikubo, J. Jacobs, T. J. Seok, K. Yu, N. Quack, C. K. Kim, R. S. Muller, and M. C. Wu, "32 × 32 silicon photonic mems switch with gap-adjustable directional couplers fabricated in commercial cmos foundry," *Journal of Optical Microsystems*, vol. 1, no. 2, p. 024003, 2021.