Optalysys

November 2026

Compute-in-Transit

Direct-drive photonics for compute-enabled interconnects

12B Platform, New Station St, Leeds LS1 4JB, United Kingdom
+44 113 426 6406 | info@optalysys.com | www.optalysys.com



1. Introduction 4
2. Background 5
3. Architecture overview 6
4. Data path 7
5. Supported operations 10
6. Implementing functions 11
7. Interconnect macro and scaling 13
8. Implementation status 14
9. Intellectual property 14

Appendix A. Technical details 15

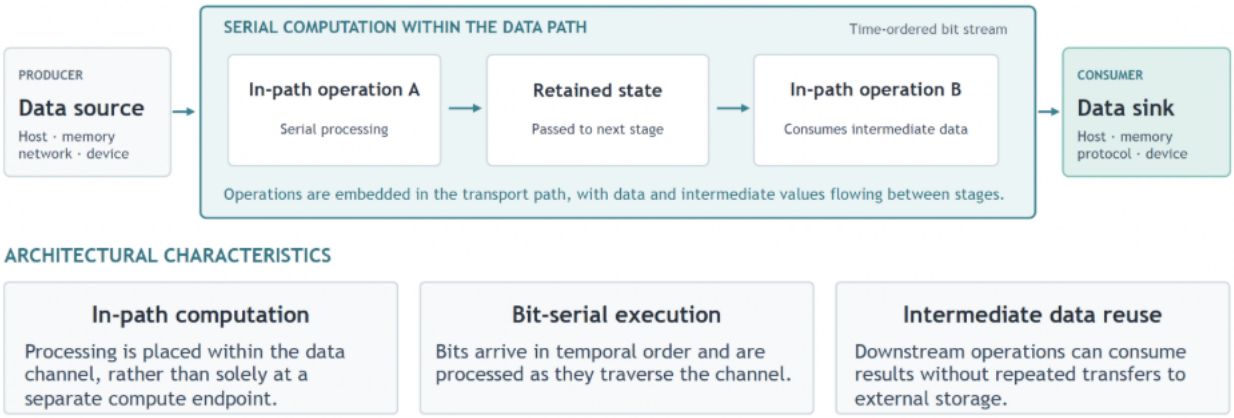
Optalysys

Inference times are becoming increasingly limited by the amount of data that must be frequently moved between memory and processors. To help meet the increasing demands, optical interconnects are being deployed to replace the conventional copper-based links, allowing higher bandwidths to be achieved at greater efficiencies.

Optalysys have developed an analog processing technology that directly integrates into the optical data path to allow math functions to be performed whilst the data is in motion. Since the data has already been converted into an optical format, the processing is conducted with minimal additional energy to the optical link budget.

The photonic technology is proven in silicon, programmable and configurable. The current design provides **2.68pJ/bit** (excl. laser) at 32 GSps per lane; future designs will compute at an efficiency of **215TOPS/W** at 112Gsps per lane and beyond.

The technology may be applied in the context of both communication and processing, creating a maths toolbox that can operate on serial streamed data at the line rate of the optical path.



1. Introduction

This paper describes the linear-drive physical layer (PHY) technology developed by Optalysys, which performs arithmetic functions on data whilst it is transmitted over an optical link. The PHY is implemented in the physical medium attachment (PMA) sublayer and operates in one of two modes: In **interconnect** mode it behaves as a conventional transceiver and is backward compatible with existing transceiver solutions. In **compute** mode the same optical and analog hardware, together with additional custom logic, performs arithmetic on the transmitted data. Figure 1 shows the block diagram of the PHY.

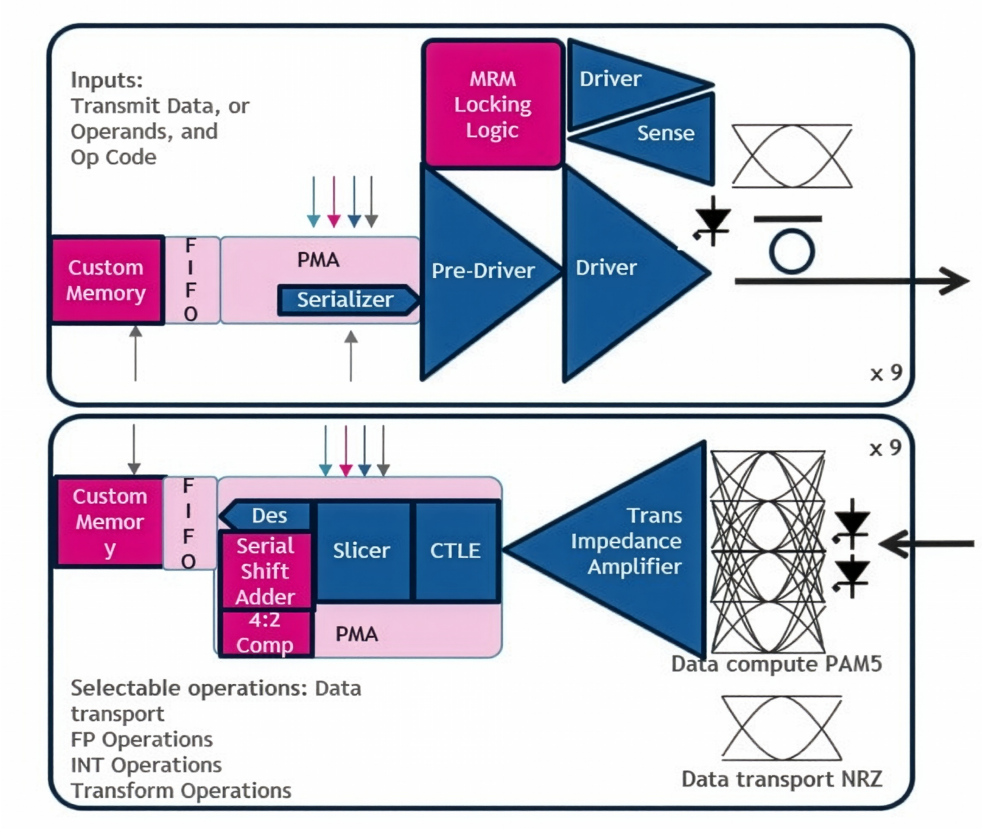


Fig.1 | Block diagram of the Optalysys linear-drive PHY technology integrated on the PMA. Blocks shown in magenta are Optalysys custom logic. The transmit and receive paths are each replicated for the nine wavelength channels.

The arithmetic is based on a four-input radix-4 Optical Fourier Transform (OFT), which is computed in the optical and analog domain using intensity modulation and direct detection. Each four-input group is equivalent to 12 radix-2 integer operations. Custom mixed-signal logic reconstructs word-level results from the detected values. The supported operations include the Fast Fourier transform (FFT), the Number Theoretic Transform (NTT), modular multiplication and reduction, modular multiply-accumulate and multiply-subtract, four-variable addition and subtraction, and FP4 multiply-accumulate with FP16, FP32 or FP64 output.

The 32 GSps analog front end and the associated logic have been designed and verified with post-layout extracted circuits, and taped out on the GlobalFoundries 45nm Fotonix process. The design scales through a nine-wavelength Wavelength-Division Multiplexing (WDM) macro, which provides 576 GSps aggregate full-duplex throughput over single-mode fibre of up to 500 m. Six or twelve macros give data configurations of 3.2 or 6.4 Tbps. The architecture is covered by the patent filings and publications listed in Section 9.

2. Background

In large multi-device systems, data is transferred between devices continuously. Part of the processing applied to this data is regular and is limited by data movement more than by arithmetic complexity. Such linear operations include Fourier and number theoretic transforms, reductions across operands, and arithmetic primitives. In current systems this processing takes place after the data has arrived, on the GPU or on a separate offload engine, consuming memory bandwidth and compute resources.

In-network reduction is already implemented in switch products. The architecture described in this paper applies a related principle at the physical layer. Arithmetic is performed on each link as the data crosses it, and the result is available at the receiver without a separate processing pass.

Optalysys

Three groups of workloads correspond to the operations that the PHY supports:

- Transforms: FFT and NTT, used in signal processing
- Reductions and modular arithmetic: four-variable addition and subtraction and modular reduction, which are used in collective operations and in cryptographic kernels
- Low-precision multiply-accumulate: FP4 E2M1 multiply-accumulate with higher-precision accumulation

3. Architecture overview

A conventional optical link encodes bits as light intensity, transmits them over a waveguide or fibre and recovers them with photodiodes. The Optalysys linear-drive PHY keeps this structure and changes the receiver. In compute mode the receiver combines the detected signals so that they equal the outputs of a radix-4 Fourier transform of the four transmitted operands. A set of custom logic blocks then converts these outputs into word-level results. The interface to the PHY can be a MAC sublayer block or a customised DMA block that accesses high-speed memory registers within the PHY.

	Interconnect mode	Compute mode
Purpose	Data transfer between devices	Arithmetic on the data during transfer
Compatibility	Backward compatible with existing transceiver solutions	Same optical and analog front end hardware, with additional arithmetic logic
Receiver path	Transport photodiodes and standard decisions	Compute photodiodes, slicers and the arithmetic chain
Output	Recovered bits	FFT, NTT, modular and multiply-accumulate results

4. Data path

In compute mode the data path runs from memory at the transmitter to a computed result at the receiver. The stages are described below and summarized in Figure 2.

1. **Serialization.** Four N-bit operand words A, B, C and D are split into synchronized bit-planes, least significant bit first. Shorter unsigned words are zero-extended. A shared word-start event keeps the bits aligned
2. **Modulation.** Each bit drives a wavelength-selective micro-ring modulator, which produces a low or high optical power. Each operand uses a different wavelength, and the four wavelengths propagate together on one waveguide or fiber
3. **Wavelength separation.** Micro-ring filters at the receiver drop each wavelength and split it between a transport branch and a compute branch
4. **Optical Fourier Transform detection.** In the compute branch the light is split and routed to photodiodes that are connected in specific balanced configurations. Adding currents of the same sign gives sums, and comparing currents from opposite arms gives differences. This produces the radix-4 outputs given below. It does not require a phase-sensitive OFT device or phase locking between the inputs
5. **Level decision.** An analog multiplexer selects the transport or compute photodiodes. After transimpedance amplification and conditioning, slicers produce small signed integers
6. **Serial reconstruction.** The Serial Shift Adder adds each small result to a stored carry, outputs the least significant bit and feeds the remaining bits back. After N steps the output bits and the final carry reproduce the full weighted sum, without a growing parallel shifter
7. **Accumulation and modular arithmetic.** 4:2 compressors combine partial rows. Programmable modular arithmetic logic uses pre-stored residues, which avoids repeating the multiplications by twiddle

factors that NTT computation requires. The result is presented on a 0.5 to 2 GHz interface that standard digital IP can use

Arithmetic primitive

For one-bit serial-slots with operands A, B, C and D, the four OFT outputs are the outputs of a radix-4 discrete Fourier transform:

OUT1 = A - B + C – D
OUT2 = (A - C) + j(B - D)
OUT3 = A + B + C + D
OUT4 = (A - C) - j(B - D)

The outputs are redundant, so the receiver detects only the real parts of OUT1 and OUT3, the real part of OUT2 (which is reused for OUT4), and the imaginary part of OUT4 (which is reused for OUT2 with reversed polarity). Because each input is binary, the outputs take few values: five levels (PAM5) for OUT1 and OUT3, and three levels (PAM3) for the detected parts of OUT2 and OUT4. For example, the slot A, B, C, D = 1, 0, 1, 0 gives A+B+C+D = 2, A-B+C-D = 2, A-C = 0 and B-D = 0.

The receiver can also be made compatible with standard PAM4 or PAM3 links. In one approach the PAM5 level is saturated to PAM4 and an additional comparison restores the fifth level. In another the outputs are compressed to PAM3 and the original value is recovered by comparison with the OFT truth table. The technology works with OOK, NRZ, Manchester, PAM, PSK and BPSK signalling, and with MZM, MRM, EAM or VCSEL modulators.

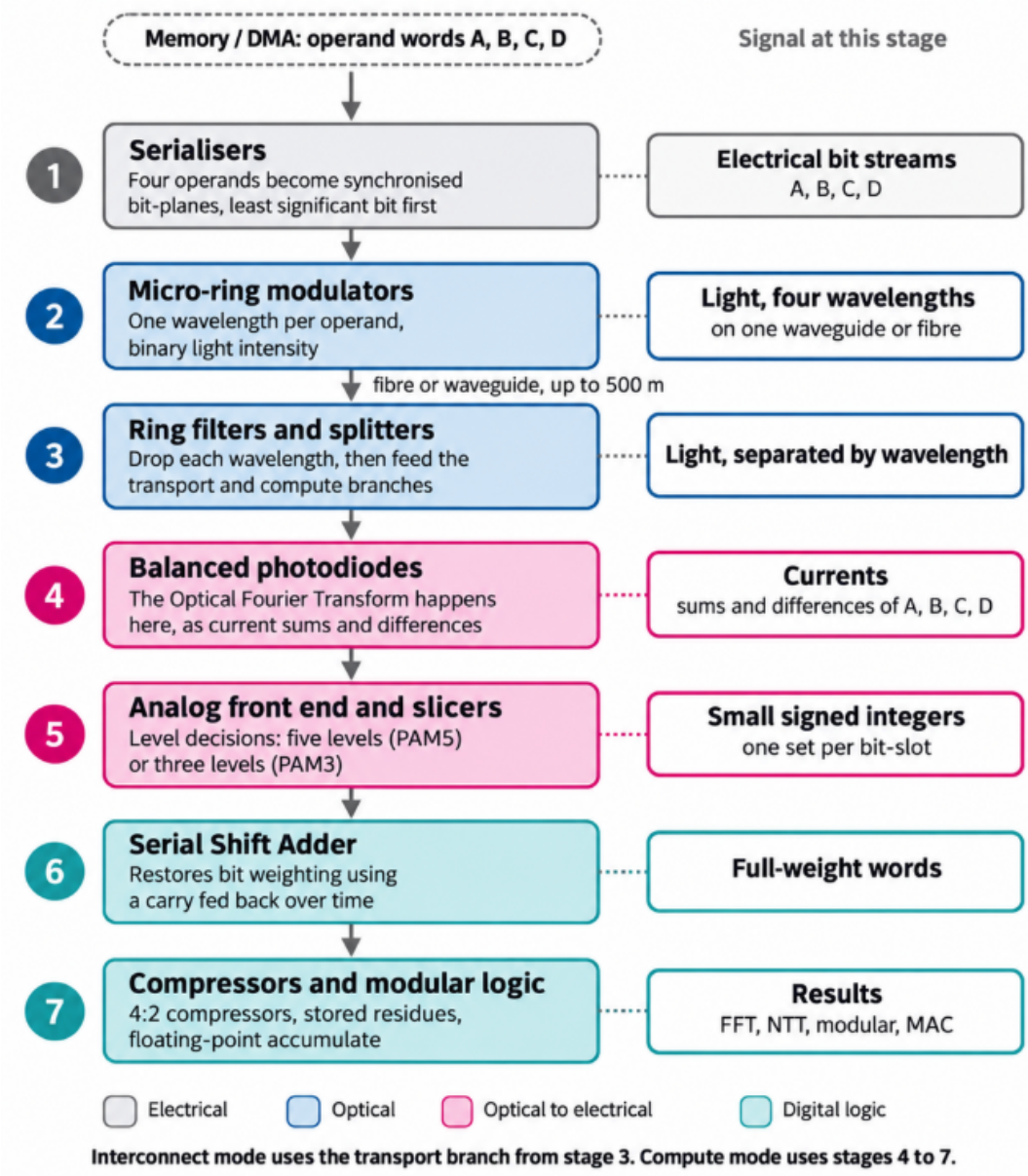


Fig. 2 | Data path in compute mode. The numbers correspond to the stages listed above.

5. Supported operations

A single OFT core is programmed with an operation code. The supported UINT32 and UINT64 operations are listed in Table 1.

Operation	Function	Inputs
FFT, NTT	Radix-4 transform of [A B C D]	A, B, C, D
ADD, SUB	A+B+C+D; A-C (or two differences at once: A-C and B-D)	A, B, C, D
ADD_SUB	Combined addition and subtraction	A, B, C, D
MOD_ADD, MOD_SUB	Modular versions of the above, modulo Q	A, B, C, D
MOD_SIM_ADD_SUB	(A+C) mod Q and (A-C) mod Q together	A, C
MOD_MUL	(A x W) mod Q	A, W
MOD_MAC, MOD_MSUB	(A x W +/- C) mod Q	A, C, W
FP4 MAC	P x Q + C in FP4 E2M1, output FP16, FP32 or FP64	P, Q, C

Table 1 | Operations supported by one core. The full list is given in the technical specification.

Latency and throughput

Table 2 gives the latency and the steady-state initiation interval for the 64-bit, 32 GSps implementation on the 45 nm GF Fotonix process, in digital cycles of the 1.5 GHz output interface

Operation	Latency	Initiation interval
FFT4	1 cycle	1 cycle
NTT4	2 cycles	1 cycle
Modular arithmetic	2 cycles	1 cycle
NTT16 (four cores)	6 cycles	1 cycle

Table 2 | Operation timing. The initiation interval assumes a filled pipeline.

Larger transforms are built from synchronized cores with high-speed memory and registers. Four FT/NTT cores form a radix-16 NTT. Throughput depends on operand length, memory and system integration.

FP4 multiply-accumulate

The FP4 multiply-accumulate operation reuses the same optical engine, Serial Shift Adder and accumulators, and uses only OUT1 and OUT3.

Shallow transmitter logic decomposes each product into Boolean partial terms, because (hP + FP/2)(hQ + FQ/2) = hPhQ + (hPFQ + hQFP)/2 + FPFQ/4. The optical engine sums or subtracts these terms over four serial channels. A floating-point accumulator then aligns the exponents, performs signed addition, and normalises and rounds the result into the selected output format. Block-scaled operands are supported by applying their scale factors consistently.

After the pipeline is filled, an intermediate result is available every two base clock cycles. With interleaved banks the design targets one multiply-accumulate per cycle of the quadrature clock, at 6.5 to 8 GHz in a 45 nm process. This is a design target and not a measured result. Efficient use of the floating-point core is under investigation.

6. Implementing functions

A function is selected by an operation code, by the assignment of operands to the four input channels, and by the processing applied to the detected quantities. This section shows how several functions are obtained from the same core. For one bit-slot the receiver provides four quantities: S = A+B+C+D, taken from OUT3; T = A-B+C-D, taken from OUT1; U = A-C, the real part of OUT2; and V = B-D, obtained from the imaginary part of OUT4, which equals D-B.

Selecting a function

Table 3 shows how operands are assigned to the input channels and which of the four quantities each function uses. The modular versions of the addition and subtraction functions use the same quantities and then reduce the result modulo Q.

Function	Operands	Quantities used	Notes
ADD	A, B, C, D	S	Sum of four operands
SUB	A, C	U	Difference of two operands
SUB_2	A, B, C, D	U, V	Two differences, A-C and B-D, at the same time
ADD_SUB	A, B, C, D	T	(A+C) - (B+D)
MOD_SIM_ADD_SUB	A, C	S, U	(A+C) mod Q and (A-C) mod Q together, with B and D set to zero

FFT	A, B, C, D	S, U, V, T	Combined as the tuple [S, U+jV, T, U-jV]
NTT	A, B, C, D	S, U, V, T	Combined with a fourth root of -1 and reduced modulo Q

Table 3 | Operand assignment and quantities used for selected functions

Example 1: addition of four operands

The operands A, B, C and D are sent least significant bit first. For each bit position i the receiver reports S(i), the sum of the four bits at that position. The Serial Shift Adder forms t(i) = S(i) + c(i), outputs the least significant bit of t(i), and passes the remaining bits on as the carry c(i+1). Table 4 shows the steps for A = 5, B = 3, C = 6 and D = 2.

Bit position	Bits A, B, C, D	S(i)	Carry in	t(i)	Output bit	Carry out
0	1, 1, 0, 0	2	0	2	0	1
1	0, 1, 1, 1	3	1	4	0	2
2	1, 0, 1, 0	2	2	4	0	2
3	0, 0, 0, 0	0	2	2	0	1

Table 4 | Serial reconstruction of A+B+C+D for A = 5, B = 3, C = 6 and D = 2.

The four output bits are zero and the final carry is 1 at weight 2^4, so the result is 16, which equals 5 + 3 + 6 + 2. The same steps apply to T. The slot values of A-B+C-D are 0, -1, 2 and 0. The carry takes the value -1 after the second step, and the output bits are 0, 1, 1 and 0, which is 6.

Example 2: radix-4 FFT and NTT

For the same operands the reconstructed quantities are S = 16, T = 6, U = -1 and V = 1. The FFT is the tuple [S, U+jV, T, U-jV], which is [16, -1+j, 6, -1-j]. For an NTT with modulus Q = 17, the value w = 4 satisfies w^2 = 16, which is -1 modulo 17. The NTT is the tuple [S, U+wV, T, U-wV] modulo Q, which is [16, 3, 6, 12].

The two transforms use the same detected quantities. Only the processing after detection differs. For the FFT, U and V are combined as

a complex number. For the NTT, the modular arithmetic logic combines them using w and reduces each result modulo Q.

Example 3: modular multiply-accumulate

To compute (A x W + C) mod Q, the modular arithmetic logic holds the residues r(i) = W x 2^i mod Q in memory. For each bit a(i) of A, taken least significant bit first, it adds r(i) to a running result z if a(i) is 1, and reduces z modulo Q. Table 5 shows the steps for A = 5, W = 3 and Q =17.

Bit position	Bit a(i)	Stored residue r(i)	Running result z
0	1	3	3
1	0	6	3
2	1	12	15
3	0	7	15

Table 5 | Modular multiplication of A = 5 by W = 3, modulo 17, using stored residues.

The product A x W reduces to 15. Adding C = 6 gives 21, which reduces to 4 modulo 17. Modular multiply-subtract follows the same steps with C subtracted. Because the residues are stored, the multiplication by W does not have to be repeated for each use of the same twiddle factor in an NTT.

7. Interconnect macro and scaling

The nine-wavelength WDM macro is the basic scaling unit. Using the same hardware, it can operate as a data communication subsystem over single-mode fiber of up to 500 m, or as a serial compute subsystem. It has nine transmit and nine receive channels: four data channels, four sideband channels and one clock channel, which together form one synchronized compute core.

- At 32 GSps, one macro provides 576 GSps aggregate full-duplex throughput
- Six or twelve macros give full-duplex data configurations of 3.2 or 6.4 Tbps
- A 112 GSps configuration, with a higher per-lane rate and fewer lanes, is planned. At this rate offset, inter-symbol interference,

jitter and path matching constrain the design more tightly, and quarter-rate operation requires 28 GHz clocks. This configuration has not been taped out

Compute throughput depends on which operations are run and on how the macros are integrated.

8. Implementation status

Table 6 summarizes the status of each part of the design

Item	Status	Notes
32 GSps analog front end and OFT receive chain	Taped out	GF 45 nm Fotonix; verified with post-layout extracted circuits
Serial Shift Adder, 4:2 compressor, modular arithmetic and FT/NTT logic	Designed; verified post-layout	Detailed specification available under NDA
112 GSps analog front end	Planned	Higher per-lane rate; tighter timing and level margins
FP4 multiply-accumulate	Principles defined	Efficient use of the floating-point core is under investigation

Table 6 | Implementation status.

9. Intellectual property

The architecture is covered by the following patent filings and publications. Further filings are pending.

- WO2026057668A1, Data transfer and processing (19 March 2026)
- WO2025168825A1, High or extended data rate optical Fourier transform method and apparatus (14 August 2025)
- EP4490603A1, Optical chiplet (15 January 2025)
- EP4490602A1, Optical encoders (15 January 2025)
- WO2025168827A1, Dense data packing and transceiver circuit for optical Fourier transform using integrated photonics (14 August 2025)

- WO2026057669A1, Arbitrary size and precision Fourier transforms via optical Fourier transforms (19 March 2026)
- WO2026057670A1, Arbitrary size and precision number-theoretic transforms via optical Fourier transforms (19 March 2026)
- WO2023187420A1, Memory architecture (5 October 2023)
- EP4677399A1, Optical Fourier transform device (14 January 2026)

Related presentation: I. Kundu and F. Bergamaschi, Hardware and Software Co-design of Fully Homomorphic Encryption accelerators using integrated photonics, FHE.org Hardware Day 2025 (25 September 2025).

Appendix A. Technical details

Serial Shift Adder

For signed input xi at bit position i and carry ci, the carry starts at c0 = 0. The adder computes ti = xi + ci, outputs the least significant bit of ti, and keeps the remaining bits (an arithmetic right shift) as c(i+1). The final carry has weight 2^N and must be retained. A word-start signal resets the carry. A stall holds the data, the bit index and the carry together. Interleaved words need separate carries.

4:2 compressor

Two full adders per bit reduce four partial rows to two while preserving the exact sum. The sum output keeps the current bit weight. The carry and Cout outputs have twice that weight. A final carry-propagate adder resolves the rows. Quarter-rate phases preserve the significance of successive slots.

Modular reduction with stored residues

For a modulus q > 1, the logic stores ri = W x 2^i mod q and a running result z. It starts with r0 = W and z = 0. For each bit ai of the other operand, taken least significant bit first, it adds ri to z if ai = 1 and reduces the sum modulo q. It advances r by doubling and conditionally subtracting q. Each addition needs at most one subtraction of q. Setting W = 1 gives plain reduction. For signed words the sign bit has weight -

2^(N-1), and a final carry c from the Serial Shift Adder contributes (2^N c) mod q.

Radix-4 FFT and NTT

The reconstructed quantities are S = A+B+C+D, T = A-B+C-D, U = A-C and V = B-D. The FFT tuple is [S, U+jV, T, U-jV]. For a modulus Q with a fourth root w that satisfies w^2 = -1 mod Q, the radix-4 NTT is [S, U+wV, T, U-wV] mod Q.

Analog front end at 32 GSps

The symbol interval is 31.25 ps. Four 8 GHz quarter-rate phases capture successive slots. An N-bit operand group needs at least N payload intervals, so the raw group rate is at most R/N for a symbol rate R, before framing and pipeline limits. A 64-slot payload takes 2 ns to serialize. Integrated heaters and a driver-sense locking loop hold the micro-ring operating points.