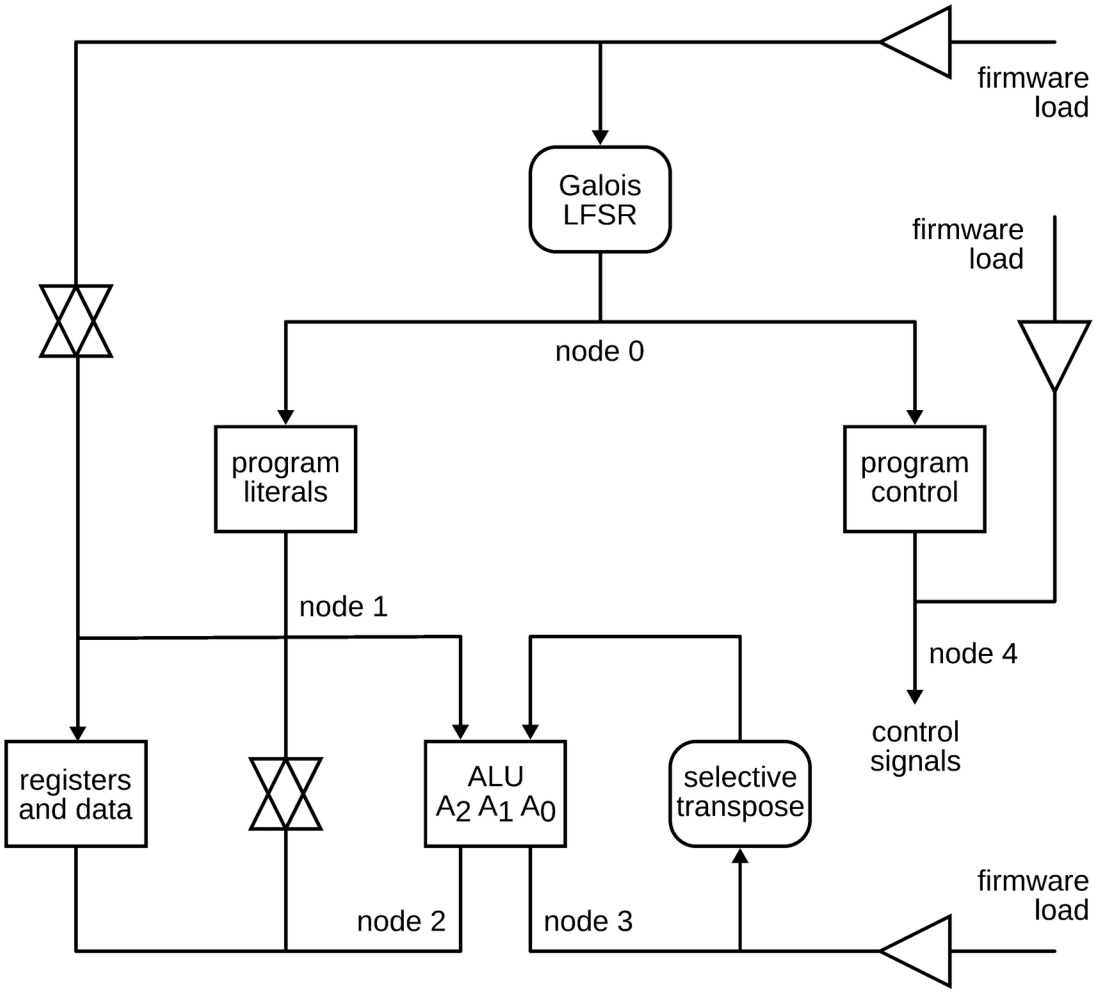
Dauug|18 Principal Data Paths



All paths in this drawing are 18 bits wide.

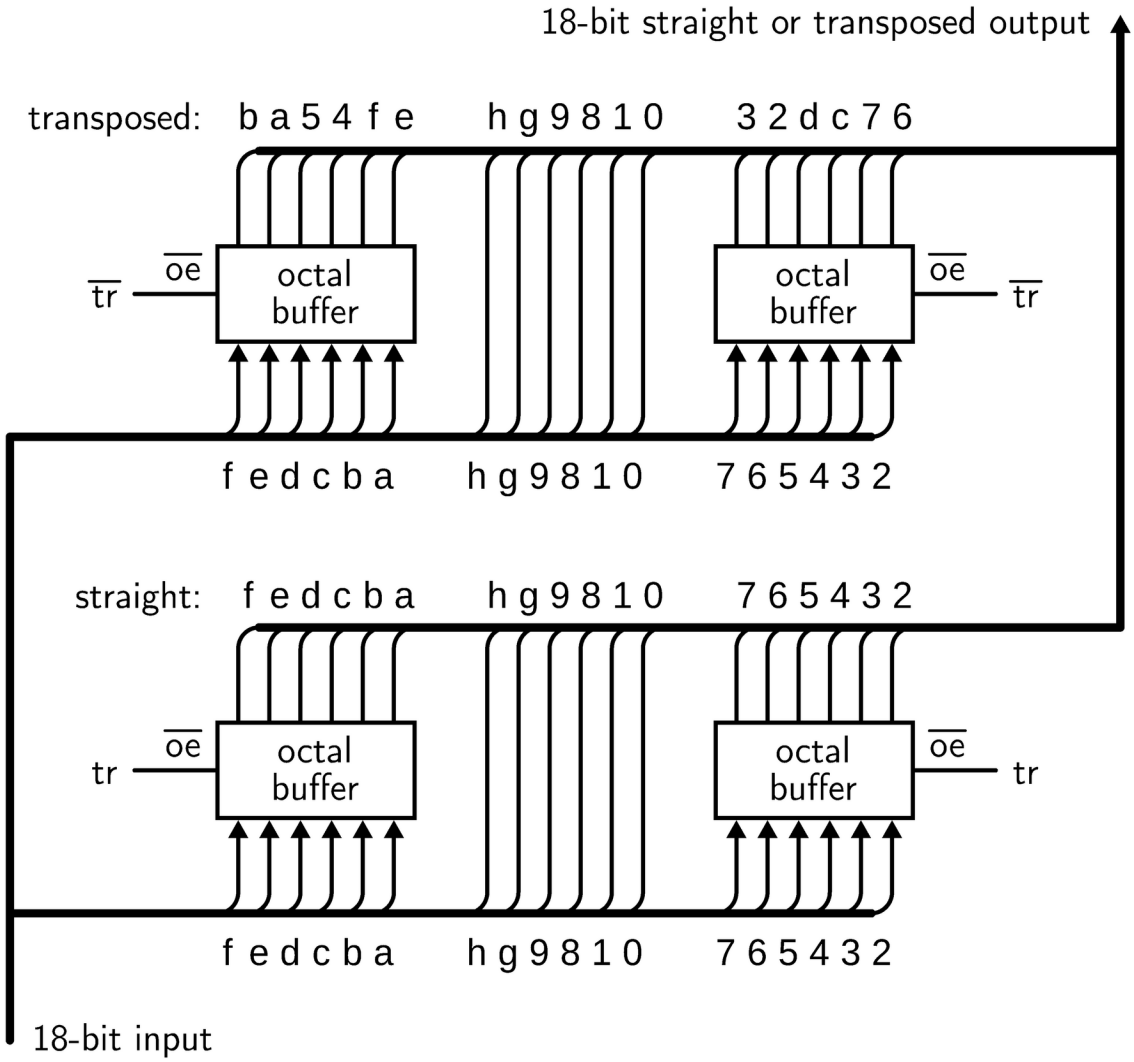
Rectangles are 256Ki×18 SRAMs with output enable.
Triangles are flip-flops with output enable.
Double triangles are bus transceivers (reversible buffers) with output enable.
Rounded rectangles are simple but non-straight-through circuits.

Node	Purpose
0	instruction pointer
1	ALU left operand, etc.
2	ALU output, etc.
3	feeds ALU right operand
4	control signals

Principal Components

Qty.	Part	Description	Location
3	GS840Z18CGT-250I	SRAM	ALU (writes normally disabled)
1	GS840Z18CGT-250I	SRAM	program literals (w.n.d.)
1	GS840Z18CGT-250I	SRAM	program control (w.n.d.)
1	GS840Z18CGT-250I	SRAM	registers and data memory
4½	SN74AUC16374	dual 8-bit flip-flop	firmware load
1½	SN74AUC16374	dual 8-bit flip-flop	Galois LFSR
3	SN74AUC16245	dual 8-bit bus transceiver	adjoining node 1
2	SN74AUC16244	dual 8-bit buffer	selective transpose
½	SN74AUC2G86	dual XOR	Galois LFSR

Selective Transpose



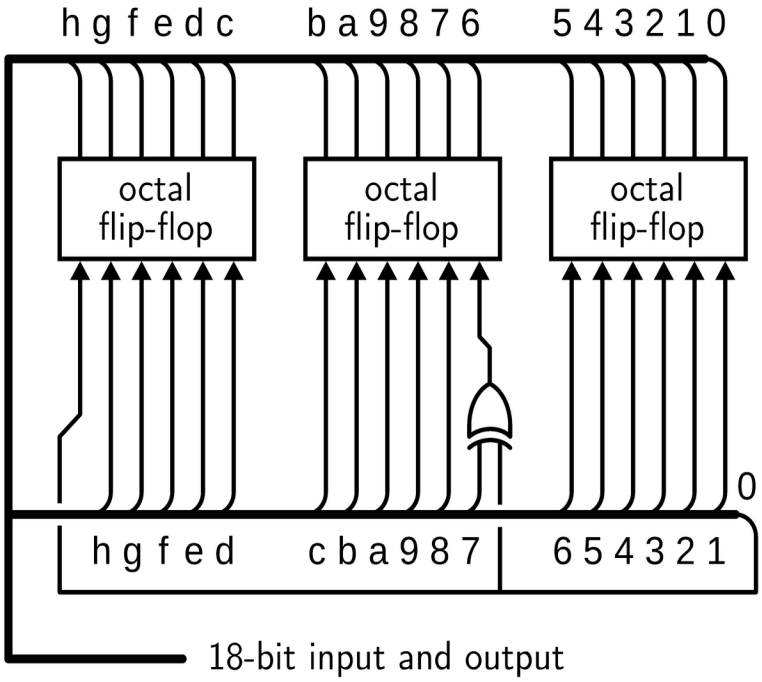
The Dauug|18 ALU has two 18-bit outputs, left and right, that each receive 6 bits from A_2, A_1, A_0. The left output feeds back to the left ALU operand, memory address, or memory data. The right output feeds the right ALU operand, in either “straight” or “transposed” bit order. The above diagram shows how this order is selected. (If you’re curious, each octal buffer has two unused bits.)

The transposition is as follows. Given a word with bit positions `h g f e d c b a 9 8 7 6 5 4 3 2 1 0`, we can write the word in a square matrix form by giving each bit a buddy. The transposition is a reflection through the main diagonal.

normal			**transposed**		
hg	fe	dc	hg	ba	54
ba	98	76	fe	98	32
54	32	10	dc	76	10

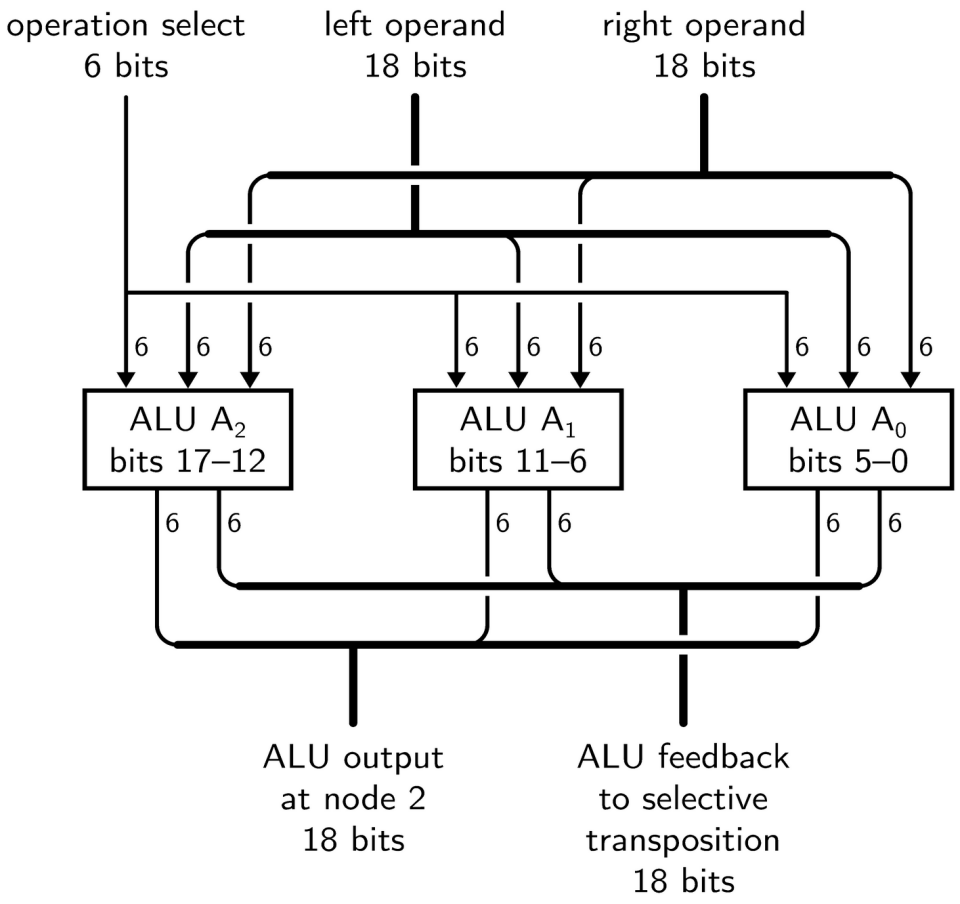
The result in word form is `h g b a 5 4 f e 9 8 3 2 d c 7 6 1 0`.

Galois Linear Feedback Shift Register (LFSR)



The Dauug|18 instruction pointer is an ordinary Galois linear feedback shift register (LFSR) with feedback taps at bits 6 and 17 and period 262,143. The main thing to note is that the flip-flop outputs can be disabled, which allows an adjacent bus transceiver to change the program counter during branch instructions. (If you're curious, each octal flip-flop has two unused bits.)

Dauug|18 Arithmetic Logic Unit (ALU)



An overview of SRAM arithmetic logic units (ALUs) is in Marc W. Abel, 2022, *A Solder-Defined Computer Architecture for Backdoor and Malware Resistance*, Ph.D. Dissertation, Wright State University, Fairborn, OH, USA, pp. 53–174. https://rave.ohiolink.edu/etdc/view?acc_num=wright167489700770166

Dauug|18 uses a bit-sliced ALU where three 6-bit subwords of an 18-bit word are evaluated simultaneously by three synchronous static RAMs (SRAMs) that contain fixed lookup tables (firmware). A 6-bit operation select identifies the function to compute on two 6-bit operands. Each SRAM produces a 6-bit “main” ALU output, such as the sum of the operands modulo 2**6, and a secondary 6-bit output, such as three copies each of the propagate and carry bits necessary to adjust the subwords for additive wraparound. For operations such as addition, both the main output and secondary output are fed back to the ALU as the next left and right operands, allowing further work such as incrementing subwords with incoming carries.

This ALU is quite robust in its capabilities. For instance, it can compute the population count (Hamming weight) of an 18-bit word in three clock cycles.