

The logo of the University of Windsor, featuring a stylized, multi-pointed star or snowflake shape with a central circular element, rendered in a light gray color.

VLSI Research Group

University of Windsor

A Hybrid DBNS Processor
for DSP Computation

G. A. Jullien, V. S. Dimitrov, B. Li, W. C. Miller, A. Lee, and M. Ahmadi



Outline

- Introduction to the Double-Base Number System (DBNS)
- Index Calculus
- Half-Index Domain Inner Product Processor
- A Hybrid DBNS Arithmetic
- A Hybrid 2-Digit Processor
- Results
- Conclusions



Introduction to the DBNS

$$h = \sum_i a_i \cdot 2^{b_i} \cdot 3^{t_i} \text{ where}$$

The binary number system is a special case (and valid member) of this representation.

	1	2	4	8	16
1					
3					
9					
27					

$$\mathbf{x} = 79$$

The canonic form of a DBNS representation is not, in general, unique.



Index Calculus

It is possible to represent any real number, with arbitrary precision, using a single non-zero digit, a_j .

The single digit can be mapped by its binary and ternary exponents, thus allowing an index calculus with which we can perform arithmetic using logarithmic-like computational units.

$h = s2^b3^t$ where $s \in \{-1, 0, 1\}$ and b, t are signed integers.

Thus $h \Rightarrow \{s, b, t\}$

As with the LNS, multiplication and division are easy.



Let $x = (s_x, b_x, t_x)$ and $y = (s_y, b_y, t_y)$ then:

$$x \cdot y = (s_x s_y, b_x + b_y, t_x + t_y) \text{ and } x/y = (s_x s_y, b_x - b_y, t_x - t_y).$$

Addition and Subtraction

$$\begin{aligned} 2^a 3^b + 2^c 3^d &= 2^a 3^b (1 + 2^{c-a} 3^{d-b}) \\ &\approx 2^a 3^b \Phi(c-a, d-b) \end{aligned}$$

$$\begin{aligned} 2^a 3^b - 2^c 3^d &= 2^a 3^b (1 - 2^{c-a} 3^{d-b}) \\ &\approx 2^a 3^b \Psi(c-a, d-b) \end{aligned}$$

$$\Phi(x, y) = 1 + 2^x 3^y \approx 2^\alpha 3^\beta \text{ and } \Psi(x, y) = 1 - 2^x 3^y \approx 2^\gamma 3^\delta$$



Half-Index Domain Processor

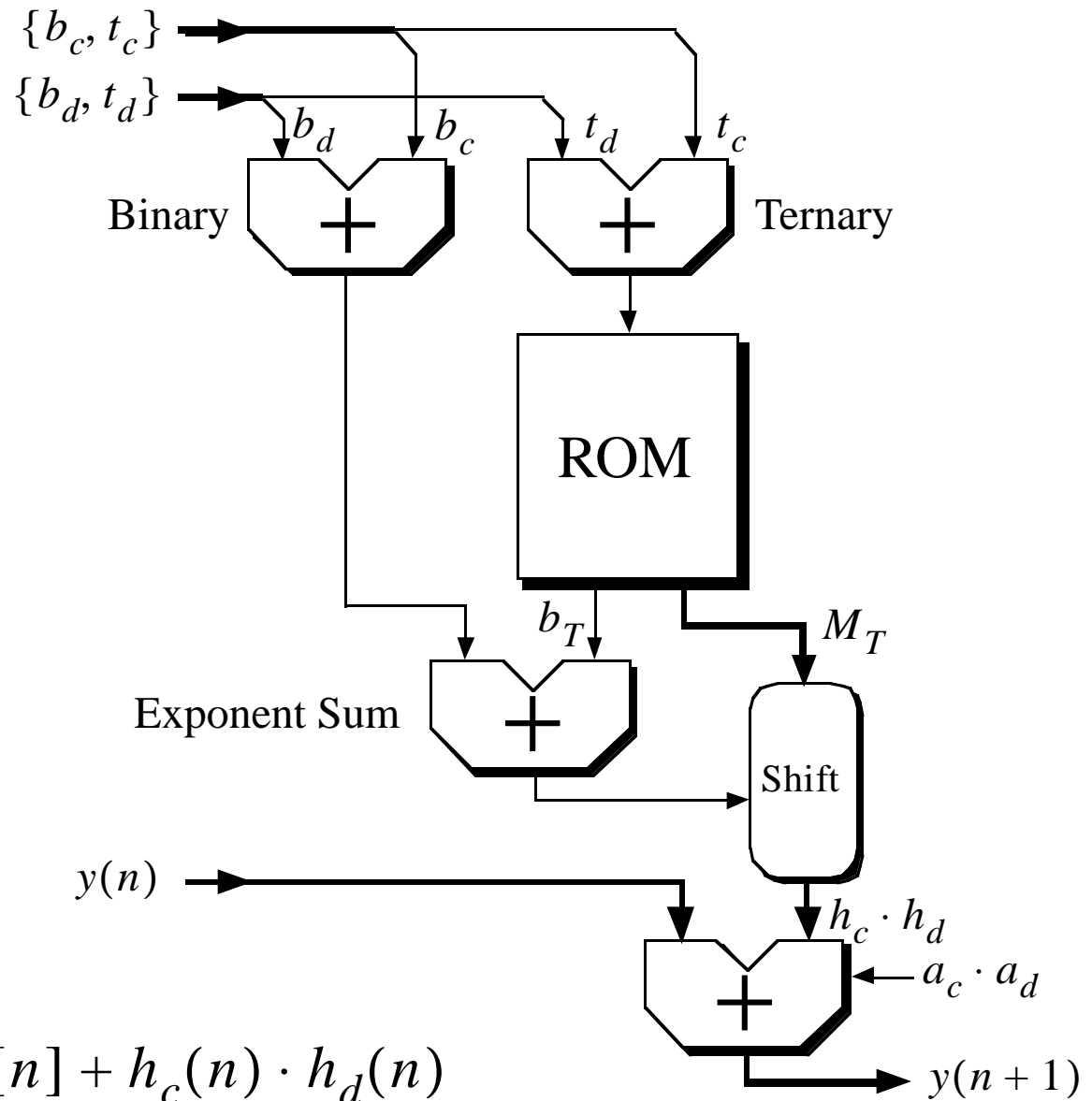
- Inner product processing targeted to DSP computations - a single multiplication in any data path.
- Only use the index representation for the multiplication, and convert to binary for the inner product accumulation.
- This avoids the addition problem and also provides the output as a binary number.
- Full precision adders are not required for the binary exponent computations.



- The binary and ternary exponents are independently summed.
- The ternary exponent is mapped to a floating point binary number

$$m_T \cdot 2^{b_T}$$

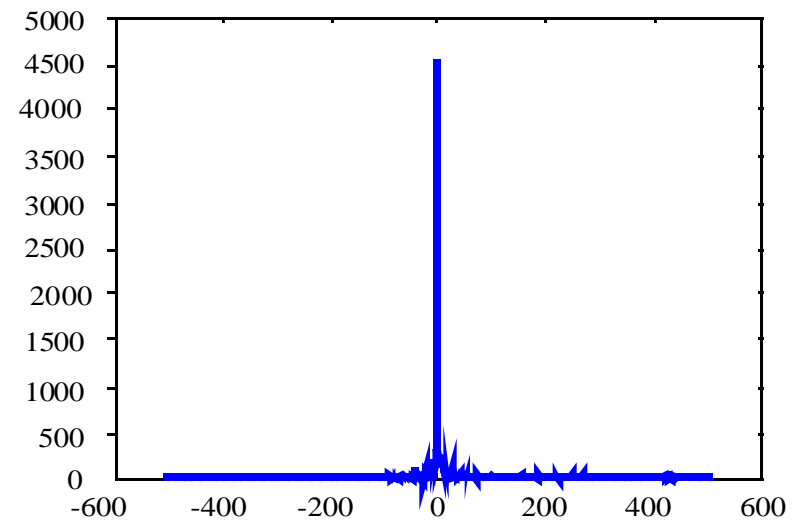
$$y[n + 1] = y[n] + h_c(n) \cdot h_d(n)$$





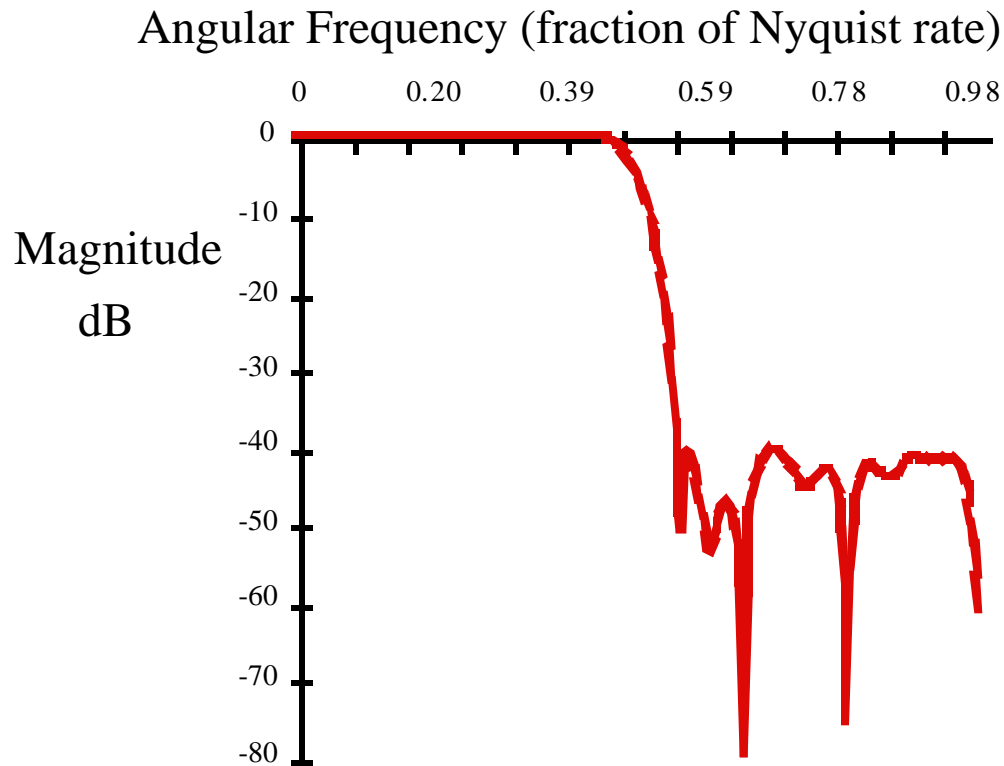
- Since the index mapping is logarithmic in each base, we experience the usual problem of having good precision for small numbers and poor precision for large numbers.
- For FIR filter coefficients, this does not turn out to be a problem since most of the coefficients are clustered at the low end of the dynamic range.

Non-Linear Representation





FIR Filter Design



#	a	b	t	#	a	b	t
0	1	-55	28	15	1	-95	56
1	-1	-273	160	16	-1	-155	85
2	-1	-260	156	17	-1	110	-73
3	1	194	-135	18	-1	-36	16
4	1	-283	171	19	1	158	-103
5	-1	-248	150	20	1	182	-125
6	-1	-280	170	21	-1	-152	93
7	1	67	-48	22	-1	23	-25
8	-1	157	-111	23	1	-264	164
9	-1	100	-68	24	1	-97	51
10	1	-45	15	25	-1	-184	114
11	1	10	-11	26	-1	-139	76
12	1	250	-169	27	1	309	-196
13	-1	-21	9	28	1	83	-53
14	-1	6	-13				

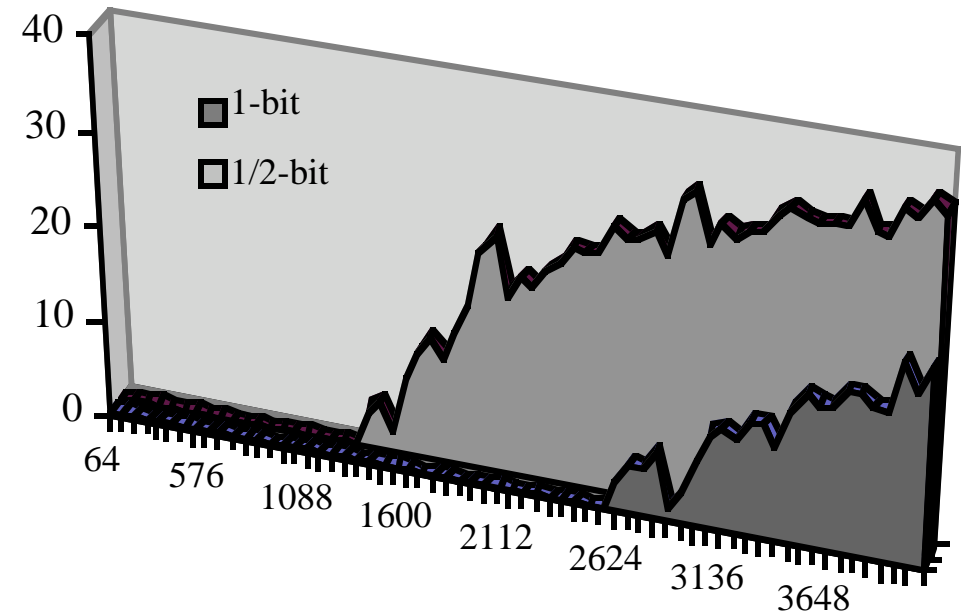
A. Lee, M. Ahmadi, G. A. Jullien, W. C. Miller, and R. S. Lashkari, "Design of 1-D FIR Filters with Genetic Algorithm", ISCAS'99, session 17.9.

Ternary exponents contained within 9-bits ($|t_c| < 256$)



A Hybrid DBNS Arithmetic

- The *data* ternary exponent requires more bits than the original binary data representation in order to maintain $\pm 1/2$ bit accuracy.
- This is a problem for the ternary conversion ROM size.
- A solution is to use a 2-digit representation only for the data (hybrid DBNS).



$\pm 1/2$ and ± 1 -bit errors for 12-bit exponent and data



2-digit representation

- $h = s_1 2^{b_1} 3^{t_1} + s_2 2^{b_2} 3^{t_2}$
- Over the entire 12-bit data $-20 \leq t_{d1}, t_{d2} \leq 28$
- Advantage:
 $|t_{di} + t_c| < 256$ therefore we can use a 9-bit ROM.
- Disadvantage: we need 2 separate channels.

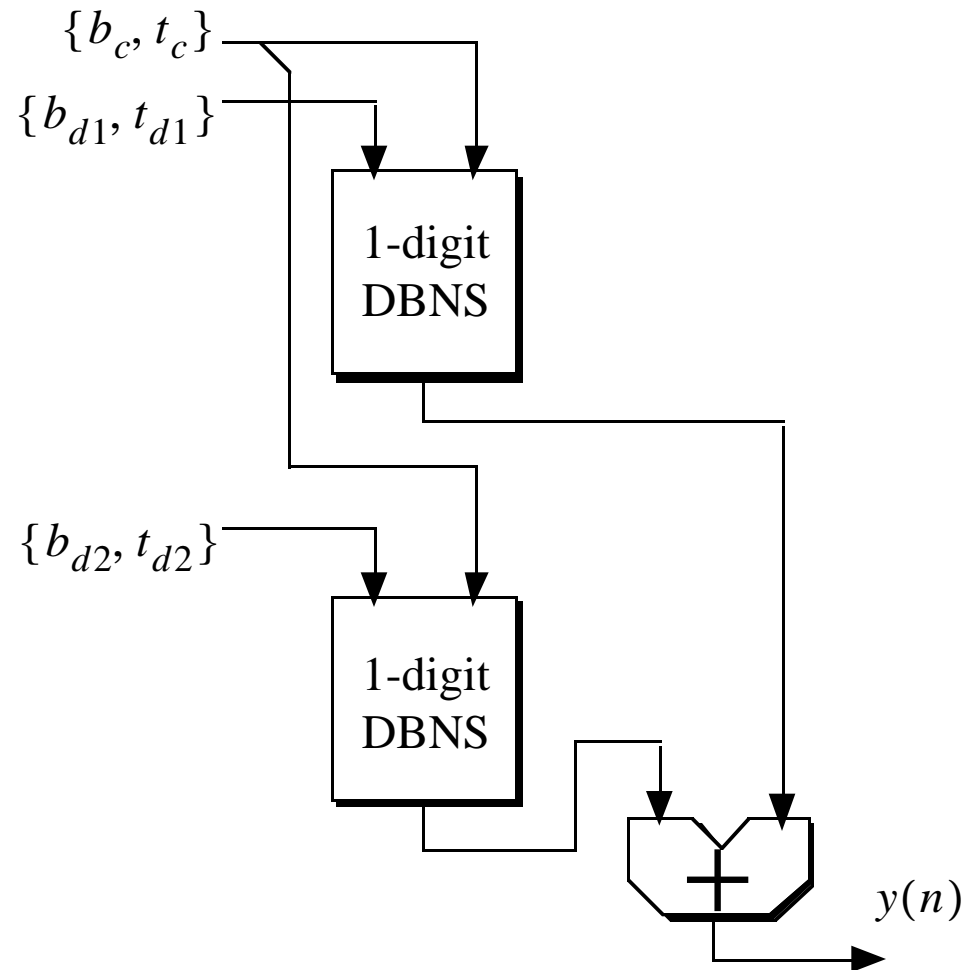
h_d	a_{d1}	b_{d1}	t_{d1}	a_{d2}	b_{d2}	t_{d2}	b_d	t_d
3568	-1	-13	14	1	-7	12	2394	-1503
3569	1	4	4	1	27	-10	2394	-1503
3570	-1	-9	12	1	9	2	-2876	1822
3571	1	-30	28	-1	-16	19	-2876	1822
3572	1	-31	27	1	25	-13	2879	-1809
3573	1	18	-4	1	29	-13	1825	-1144
3574	-1	2	5	1	28	-10	-2391	1516

Example comparisons between 1-digit and 2-digit exponents



A Hybrid 2-digit processor

- Although we have doubled the channels, each channel is 16 times smaller (based on ROM size)
- The hybrid processor is therefore 8 times smaller than the 1-digit processor.



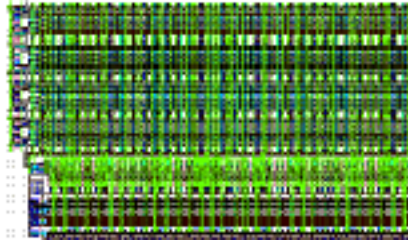


Results

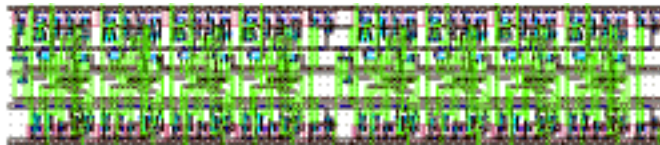
- The hybrid processor has been verified by simulation.
- The 1-digit DBNS processor has a similar structure to the Fermat ALU for which we have proof-of-concept silicon in 0.5μ and 0.35μ CMOS.
- Simulations of components for a complete 0.5μ 1-digit processor yield power savings of $>50\%$ compared to an equivalent binary processor.
- A complete 0.35μ CMOS processor for a programmable 53-tap filter is currently in the design stage. We will report on measured power and area savings in a future publication.



0.5 μ DBNS ALU Layout Components



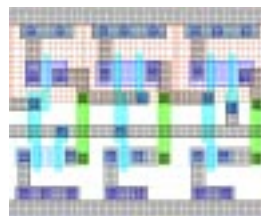
8-bit dynamic ROM



8-bit dynamic adder

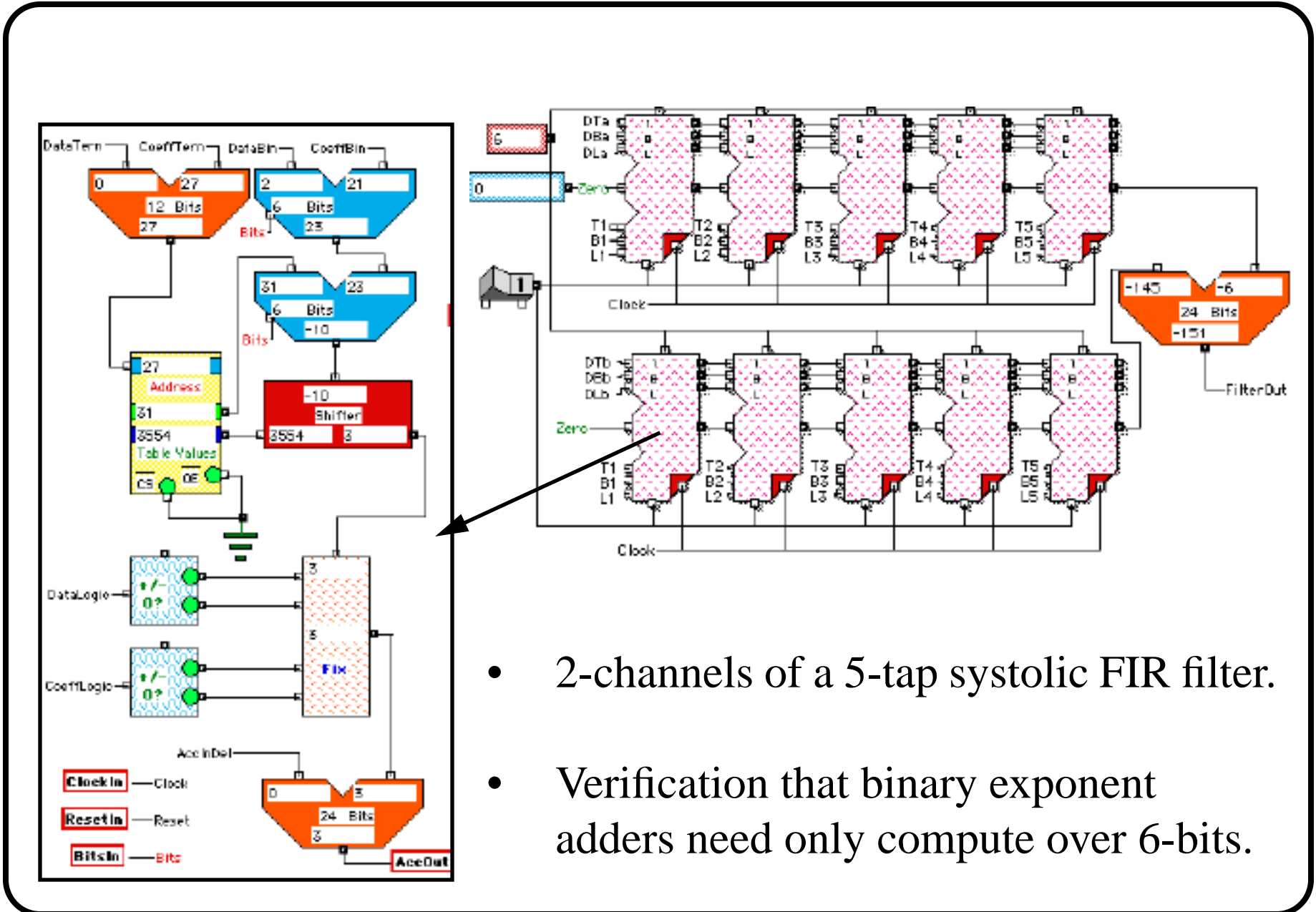


7 X 4 barrel shifter



dynamic latch

- Layout components used to estimate speed and power of 2-digit DBNS ALU.
- Circuits designed for 120MHz operation.
- Initial estimate of <20mW/100MHz
- Less than 50% of equivalent binary processor.



- 2-channels of a 5-tap systolic FIR filter.
- Verification that binary exponent adders need only compute over 6-bits.



Conclusions

- Introduced a hybrid (1-digit/2-digit) DBNS processor for FIR filter applications
- 2 independent channels required but area complexity has been reduced by 80% compared to the 1-digit processor.
- The final architecture provides more than 80% area savings over its 1-digit counterpart.
- At least 50% power savings compared to equivalent binary processor.