EECS 427
Lecture 11: Shifters / CAD5
Reading: 11.5

Shift types

- Logical
  - Shift in 0s
  - left shifts pretty much always logical
  - Left shift is multiplication by 2^n
- Rotate
  - Useful in some coding and crypto applications
  - Shift in the bits that were shifted out
  - Data preserving
  - Shifts can be rotates followed by bitwise mask

More shift types

- Arithmetic
  - Shift “in” the high bit
  - Sometimes use the carry control bit instead
  - Useful for lots of things besides the obvious divide by 2^n (and round down)
  - In particular, conditional generation of masks based on sign of value
  - Absolute value:
    - rb = ra sra ra 15
    - ra = ra xor rb
    - ra = ra - rb

Shift Examples

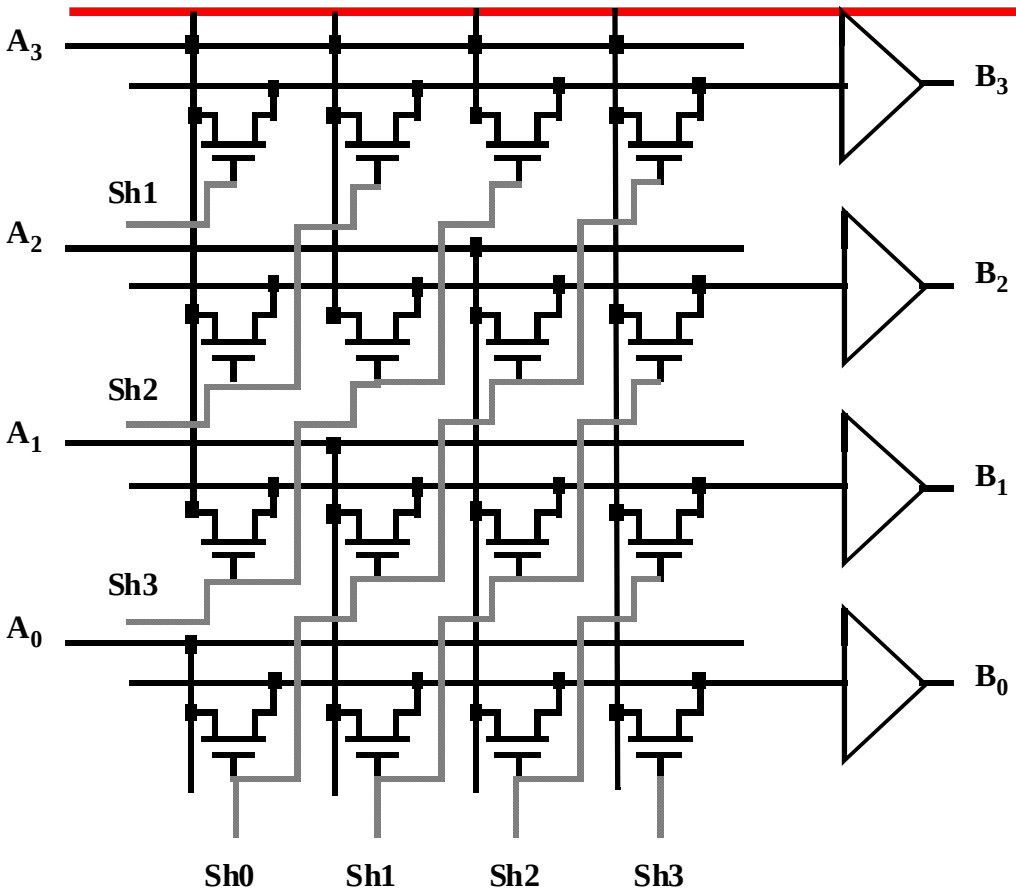
- Arithmetic vs. logical shift (Start with 1101)
  - Logical shift (baseline op for us)
    - Logical shift right by 1: 0110
    - Logical shift right by 2: 0011

    } Shift in 0s for logical right shifts
    - Logical shift left by 1: 1010
    - Logical shift left by 2: 0100

    } Shift in 0s for left shifts
  - Arithmetic shift (not baseline)
    - Arithmetic left shifts same as logical
    - Arithmetic right shift by 1: 1110
    - Arithmetic right shift by 2: 1111

    } Repeat sign of MSB for arithmetic right shifts

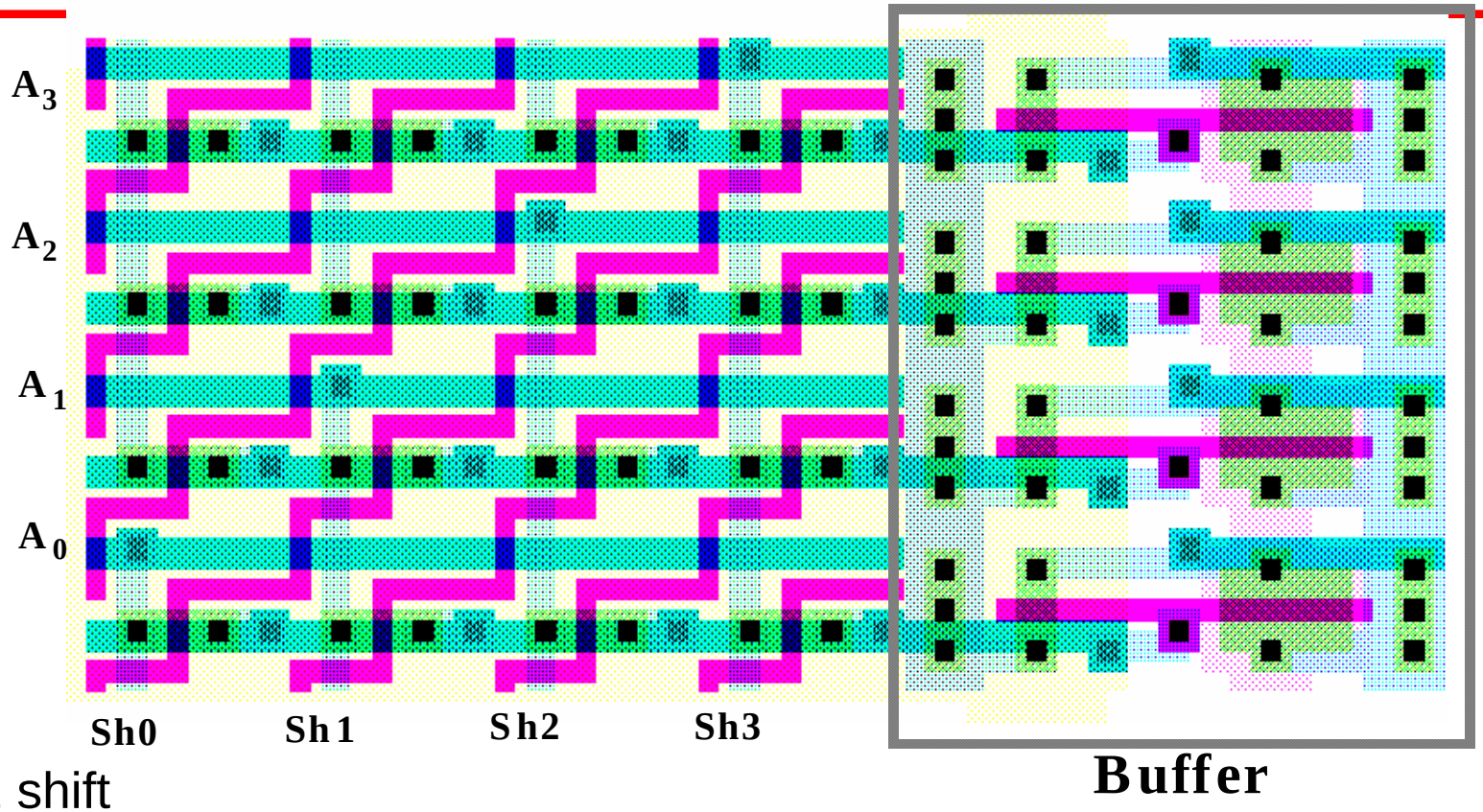
Barrel Shifter



Area Dominated by Wiring

Requires decoding of shift amount given in instruction word

4x4 barrel shifter



Only 1 shift bit high at any time

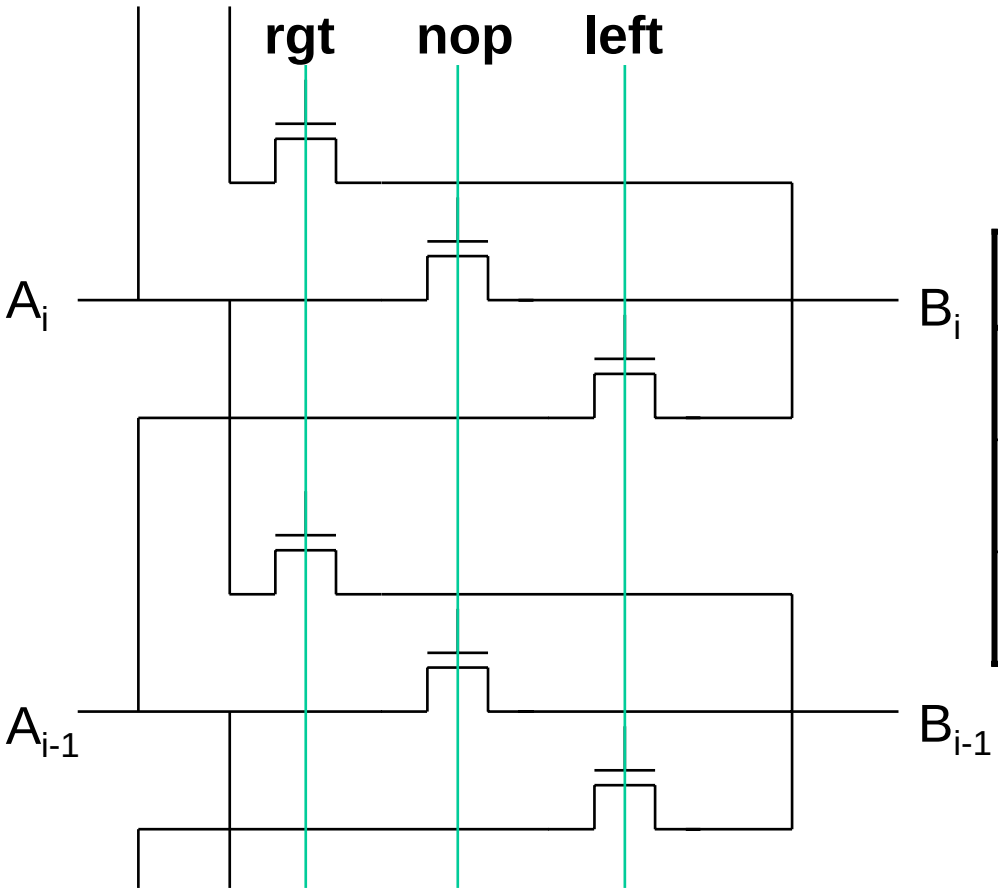
$$\mathbf{Width_{barrel} \sim 2\, p_m\, N}$$

N is max shift width (15 for us), p_m is metal pitch

Barrel Considerations

- Buffer should really have weak feedback (for full swing)
- Shown choices for horiz / diagonal / vertical lines not the only ones
- How to implement shifts in both directions?
  - This turns out to be nice for barrel because one direction only requires a triangle, and the two triangles can be fit together to form a square
- Fast for A through, but B goes through decoder
- Size becomes a problem at more bits

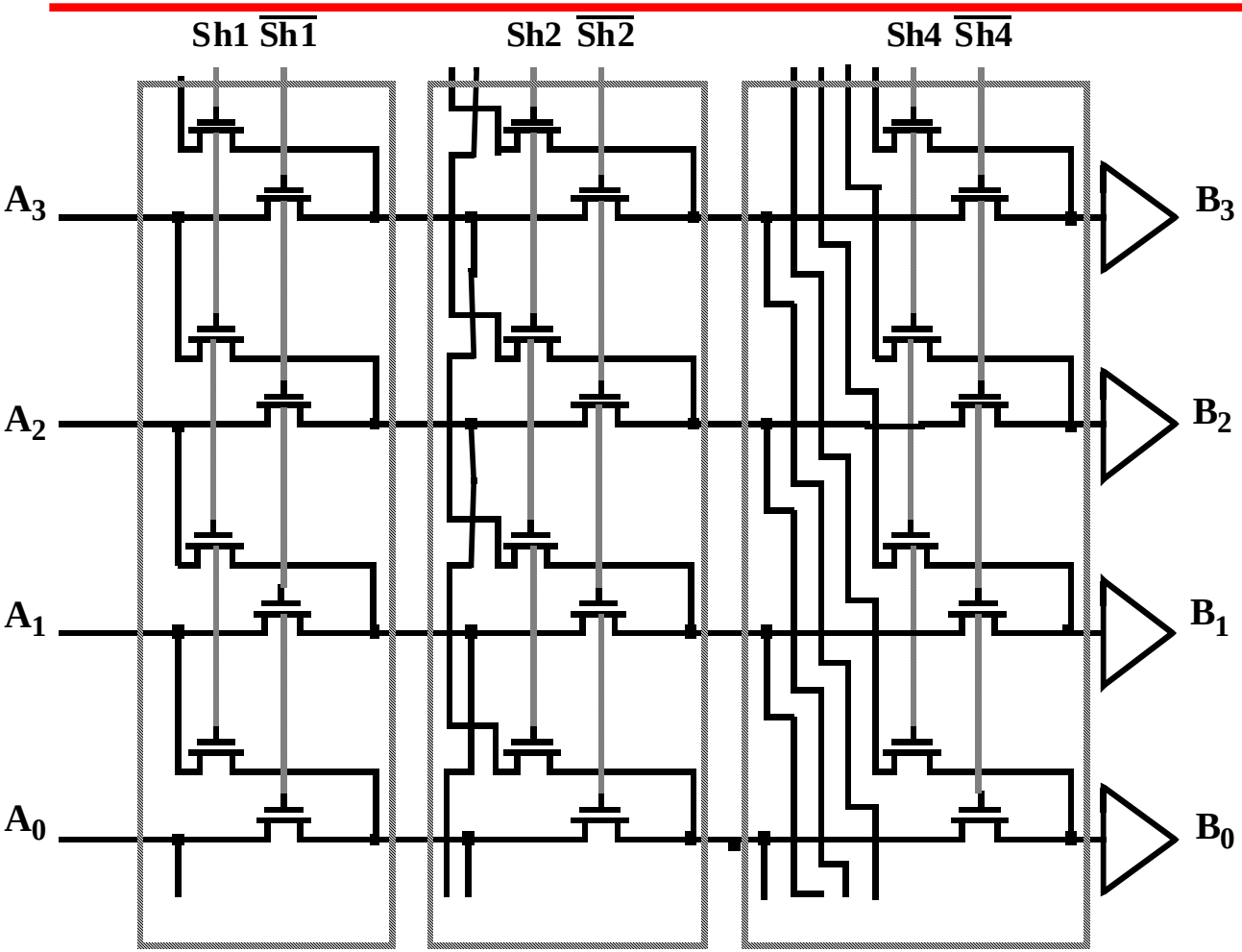
The Binary Shifter Concept



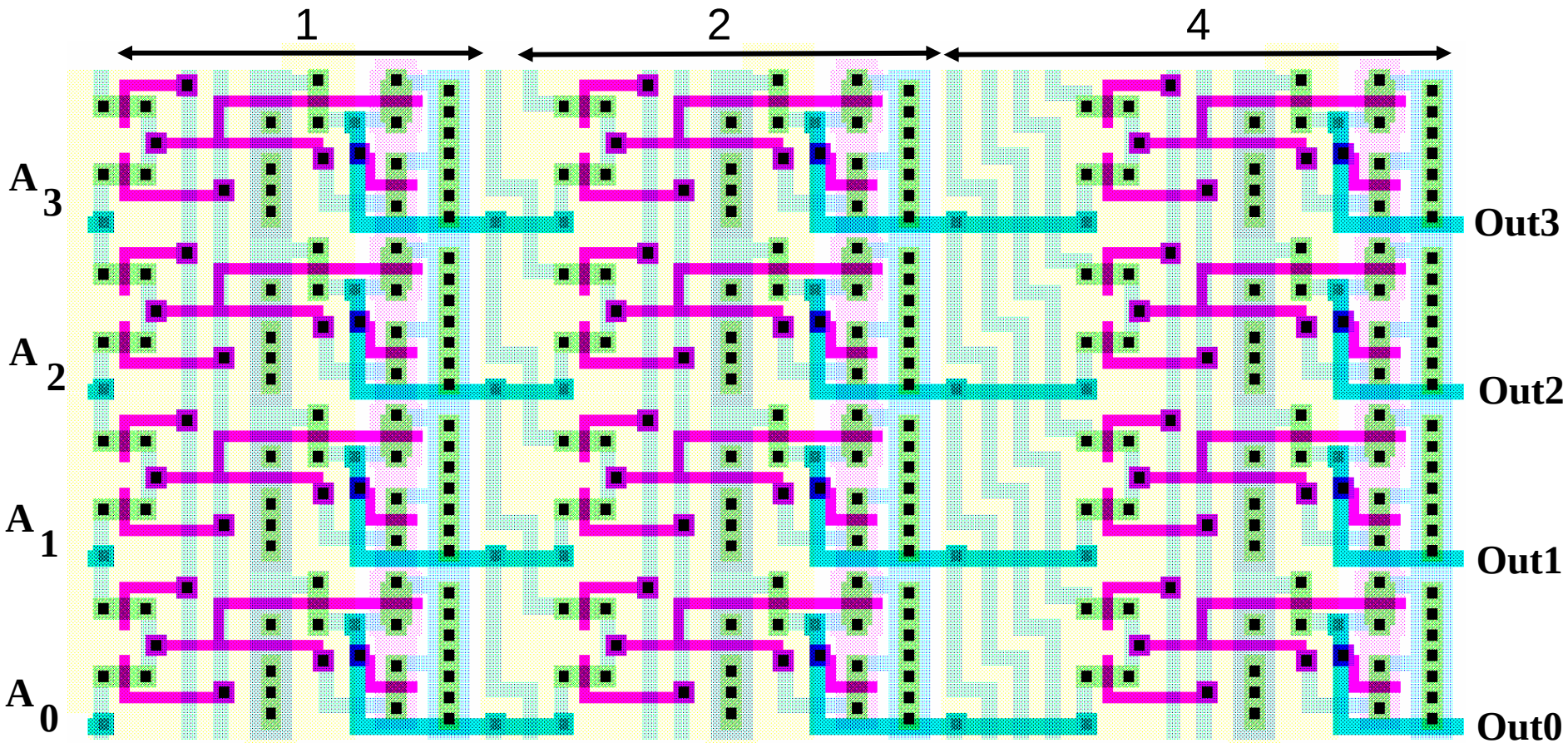
A_i	A_{i-1}	rgt	nop	left	B_i	B_{i-1}
A_1	A_0	0	1	0	A_1	A_0
A_1	A_0	1	0	0	0	A_1
A_1	A_0	0	0	1	A_0	0

Thanks to: Irwin/Narayanan

Logarithmic Shifter



0-7 bit Logarithmic Shifter



$$width_{\log} \approx p_m\left(2K + \left(1 + 2 + \ldots + 2^{K-1}\right)\right) = p_m\left(2^K + 2K - 1\right)$$

$K = \log_2 N$

Log Shifter Considerations

- Wiring pattern looks very similar to kogge-stone adder
- No decoding needed
- How to handle both directions?
  - Use binary shifters with connections in both directions?
  - Some way to avoid that?
- How to implement those small muxes?
  - Use pass transistors like barrel shifter?

Other Spins on Shifters

- Log_4 instead of log_2 (cross between barrel / log)
  - Advantage: Fewer stages of pass transistors
  - Disadvantage: must re-encode the control bits
  - Ex: 16 bits → 4 stages for log_2 vs. only 2 stages for log_4
- Rotate instruction – could be an ISA addition
  - 11010101 → rotate right by 5 bits → 10101110
  - Both shift directions can be rotations with appropriate masking
- Reverse order shifters – do the big shifts first
  - Helpful because the big shifts have larger wire capacitances
  - Elmore delay is reduced by having large caps charged through the least resistance (fewest pass transistors)

CAD5

- Logical “left shift” is only required instruction
- Shift amount could be negative though which means right shift is hidden in there too.
- Log shifter is harder. Much less uniform, and shifting in both directions is hard
- Size should be dominated by wiring, and will “want” to be narrower than your bitslice width.
- Don’t give in! Keep to your bitslice width.

Shifter Summary

- Barrel shifters are area-intensive (though at 16 bits are probably no bigger) but have only 1 pass transistor per path
  - Lots of junction capacitance though
- Log shifters are more versatile for wider data
- Lots of variations: log base, reverse order, pass transistor vs. T-gate
- Don’t forget data dependent shifts and dealing with shifts in both directions