

4.9 Shift and Add Multiplier



A Shift-and-Add Multiplier

- Problem description:
 - Design a multiplier for **unsigned** binary numbers.
 - Form the *product* $A \times B$
 - the 1st operand (A): *multiplicand*,
 - the 2nd operand (B): *multiplier*.
- *Serial-parallel multiplier*:
 - The multiplier bits are processed serially, but addition takes place in parallel.

J.J. Shann 4-78



Multiplication for Unsigned Binary Numbers

- Multiplication for *unsigned* binary numbers:

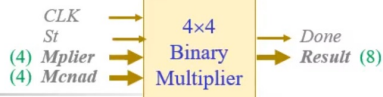
– E.g.:

Multiplicand	→	1 1 0 1	(13)
Multiplier	→	1 0 1 1	(11)
		1 1 0 1	
		1 1 0 1	
Partial products	}	1 0 0 1 1 1	
		0 0 0 0	
		1 0 0 1 1 1	
		1 1 0 1	
		1 0 0 0 1 1 1 1	(143)

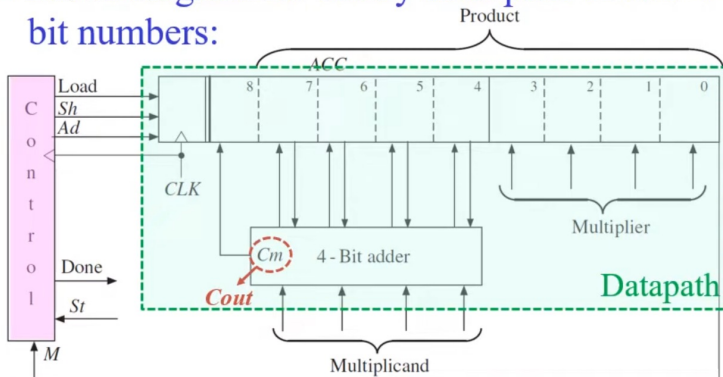
- Binary multiplication requires only *shifting* and *adding*.
- Each *partial product* is either *the multiplicand shifted over by the appropriate # of places* or *zero*.
- Each new partial product is added in as soon as it is formed.

4-79

Block Diagram of 4×4 Multiplier



- Block diagram for binary multiplier of two 4-bit numbers:



St: start signal; *M*: current multiplier bit

Load: load multiplier into the lower 4 bits of ACC and clear the upper 5 bits of ACC

Sh: shift signal, shift all 9 bits of ACC to the **right** by the next clock pulse

Ad: add signal, transfers the adder outputs to the upper 5 bits of ACC by the next clock pulse

Done: done signal

St: start signal

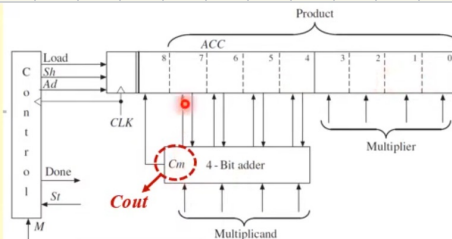
M: current multiplier bit

Load: load multiplier into ACC[3:0] and clear the ACC[8:4]

Sh: shift signal, shift ACC[8:0] to the **right** by the next clock pulse

Ad: add signal, transfers the adder outputs to the ACC[8:4] by the next clock pulse

Done: done signal



- a 4-bit multiplicand
- a 4-bit multiplier register
- a 4-bit full adder
- an 8-bit register for the product
 - The product register serves as an accumulator to accumulate the sum of the **partial product**.
 - The contents of the product register will be **shift** to the **right** each time before added with the multiplicand.
- an extra bit at the left end of the product register
 - temporarily stores any **carry** generated by the **adder**