Digital Systems Design and Laboratory
[Lab 2. Pipelined Multiplier]

TA: Sheng-Ya Lin

CSIE Department

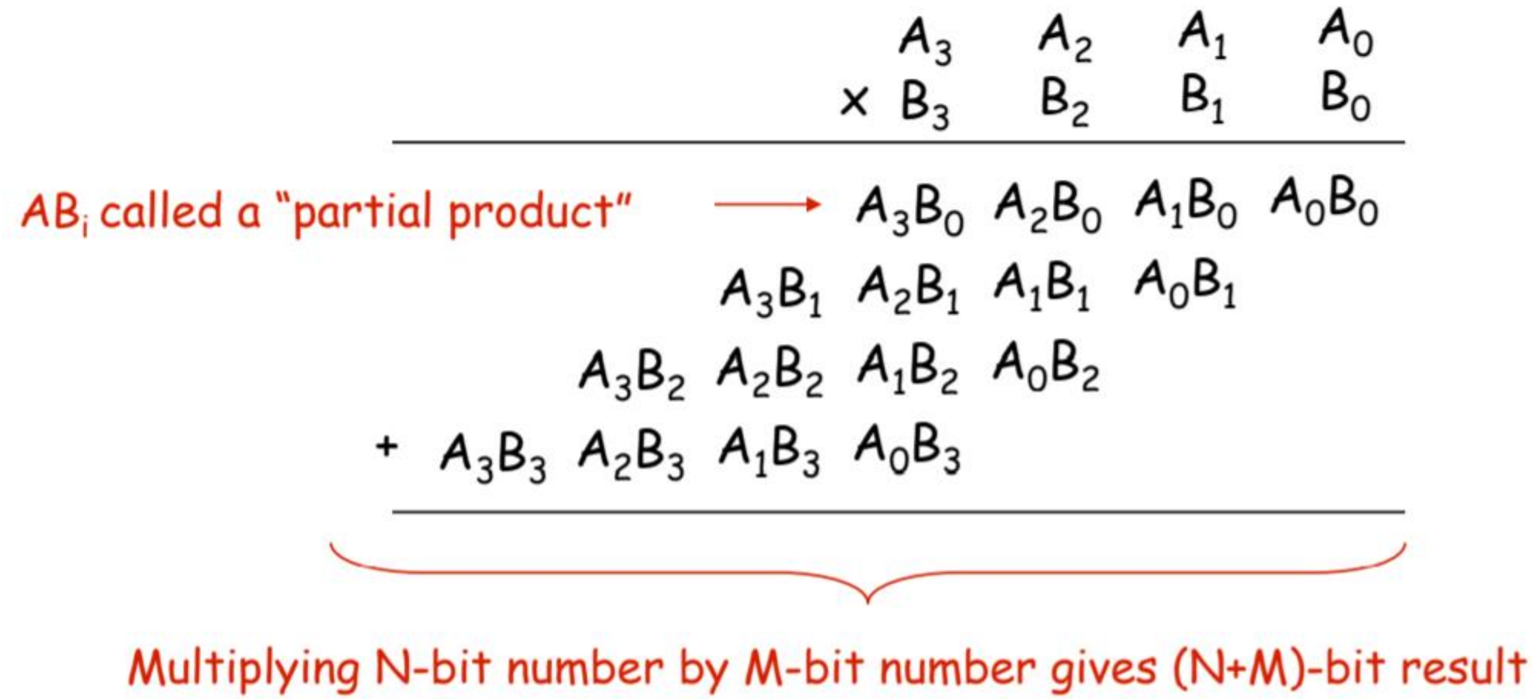
National Taiwan University

Outline

- ❑ **Multiplier**
- ❑ Pipeline
- ❑ Fast Multiplier
- ❑ Assignment
- ❑ Appendix

Definition

- ❑ Unsigned multiplier
  - ➢ Multiplicand × Multiplier = Product
  - ➢ Signed multipliers will not be covered in Lab 2



$$\begin{array}{cccccccc}
 & & & & A_3 & A_2 & A_1 & A_0 \\
 & & & \times & B_3 & B_2 & B_1 & B_0 \\
\hline
 & & & & A_3B_0 & A_2B_0 & A_1B_0 & A_0B_0 \\
 & & & A_3B_1 & A_2B_1 & A_1B_1 & A_0B_1 & \\
 & & A_3B_2 & A_2B_2 & A_1B_2 & A_0B_2 & & \\
+ & A_3B_3 & A_2B_3 & A_1B_3 & A_0B_3 & & & \\
\hline
\end{array}$$

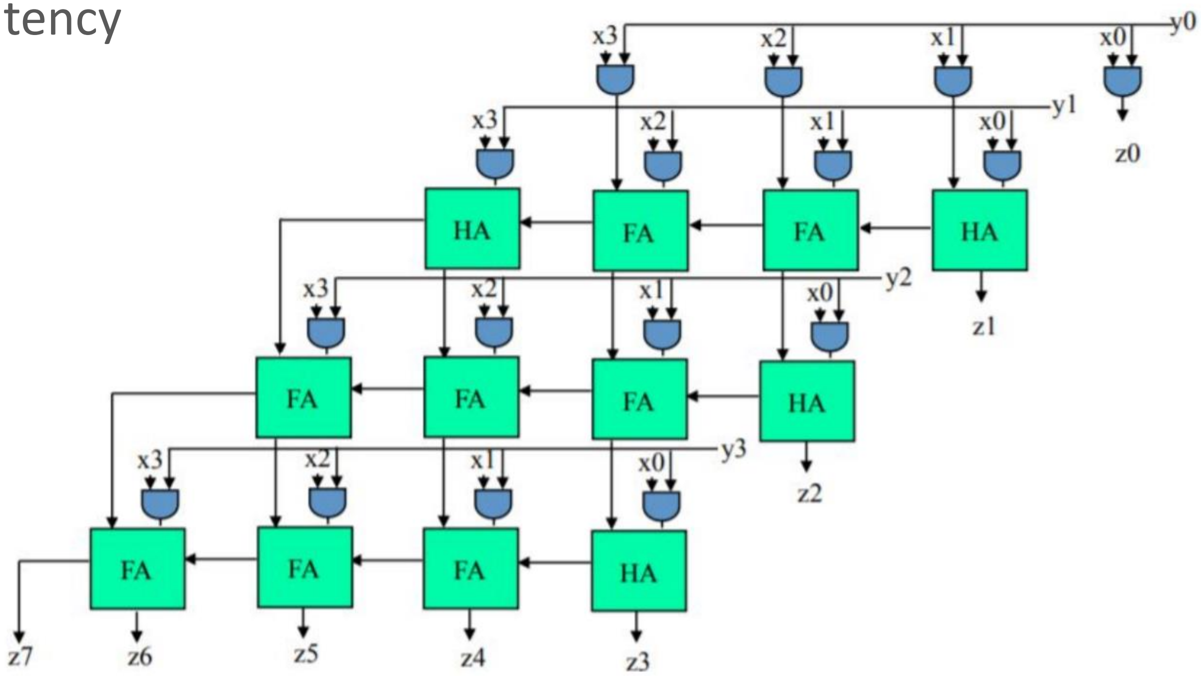
AB_i called a "partial product" →

Multiplying N-bit number by M-bit number gives (N+M)-bit result

http://courses.csail.mit.edu/6.111/f2008/handouts/L09.pdf

Metrics

- For an n-bit × n-bit multiplier
- Latency (longest propagation delay)
  - ≈ 2n*delay(FA)
- Throughput (operations per second)
  - ≈ 1/Latency



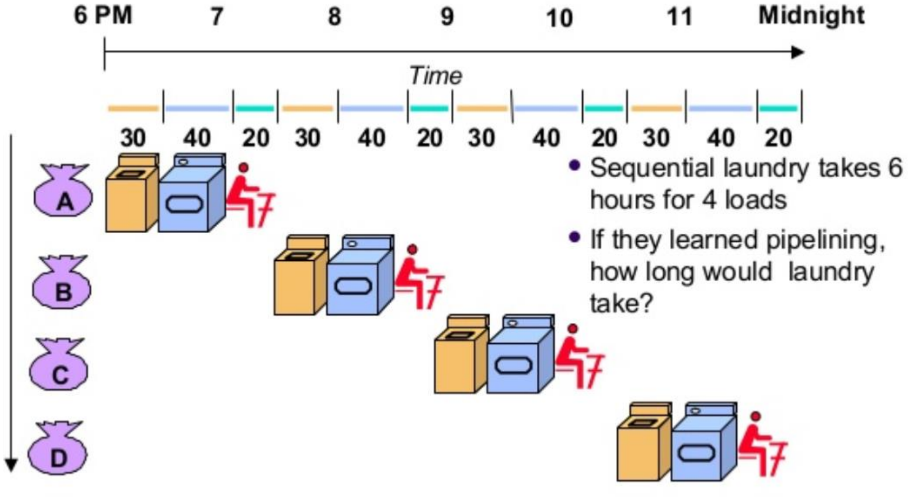
http://courses.csail.mit.edu/6.111/f2008/handouts/L09.pdf

Outline

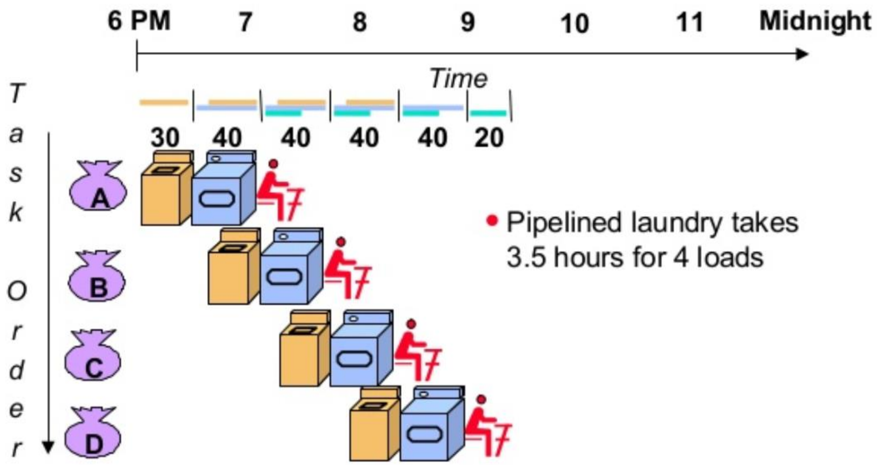
- ❑ Multiplier
- ❑ **Pipeline**
- ❑ Fast Multiplier
- ❑ Assignment
- ❑ Appendix

How to Increase Throughput?

- ❑ Cut the circuit into different stages
- ❑ Do different tasks in each stage at the same time
- ❑ Does the latency change? How about the throughput?



- Sequential laundry takes 6 hours for 4 loads
- If they learned pipelining, how long would laundry take?

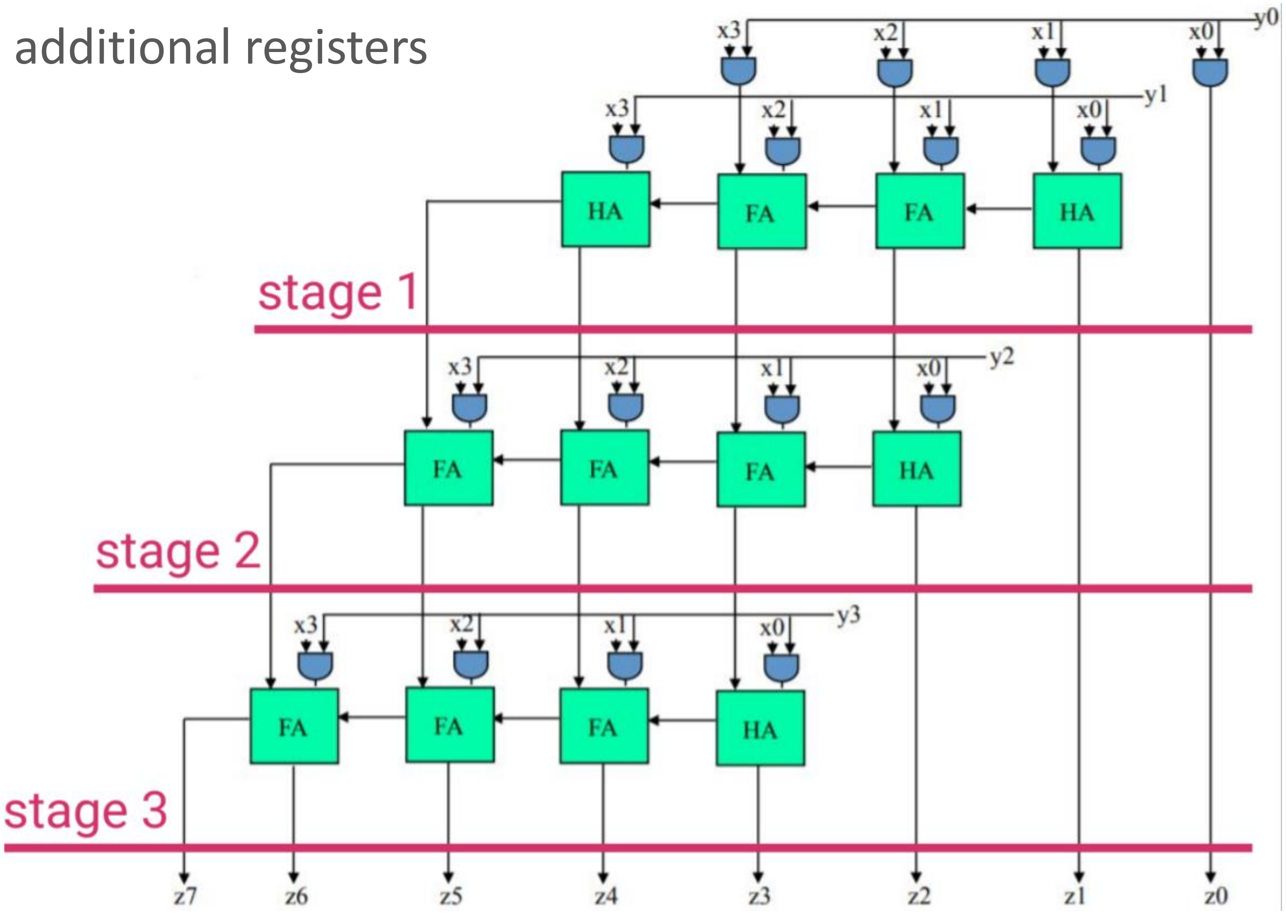


- Pipelined laundry takes 3.5 hours for 4 loads

https://www.slideshare.net/siddiqueibrahim37/pipelining-41608675

Pipelined Multiplier

- ❑ Add registers at the end of each stage to keep states
- ❑ x[3:0] and y[3:0] should also be kept in each stage
  - ➢ Use additional registers



http://courses.csail.mit.edu/6.111/f2008/handouts/L09.pdf

Outline

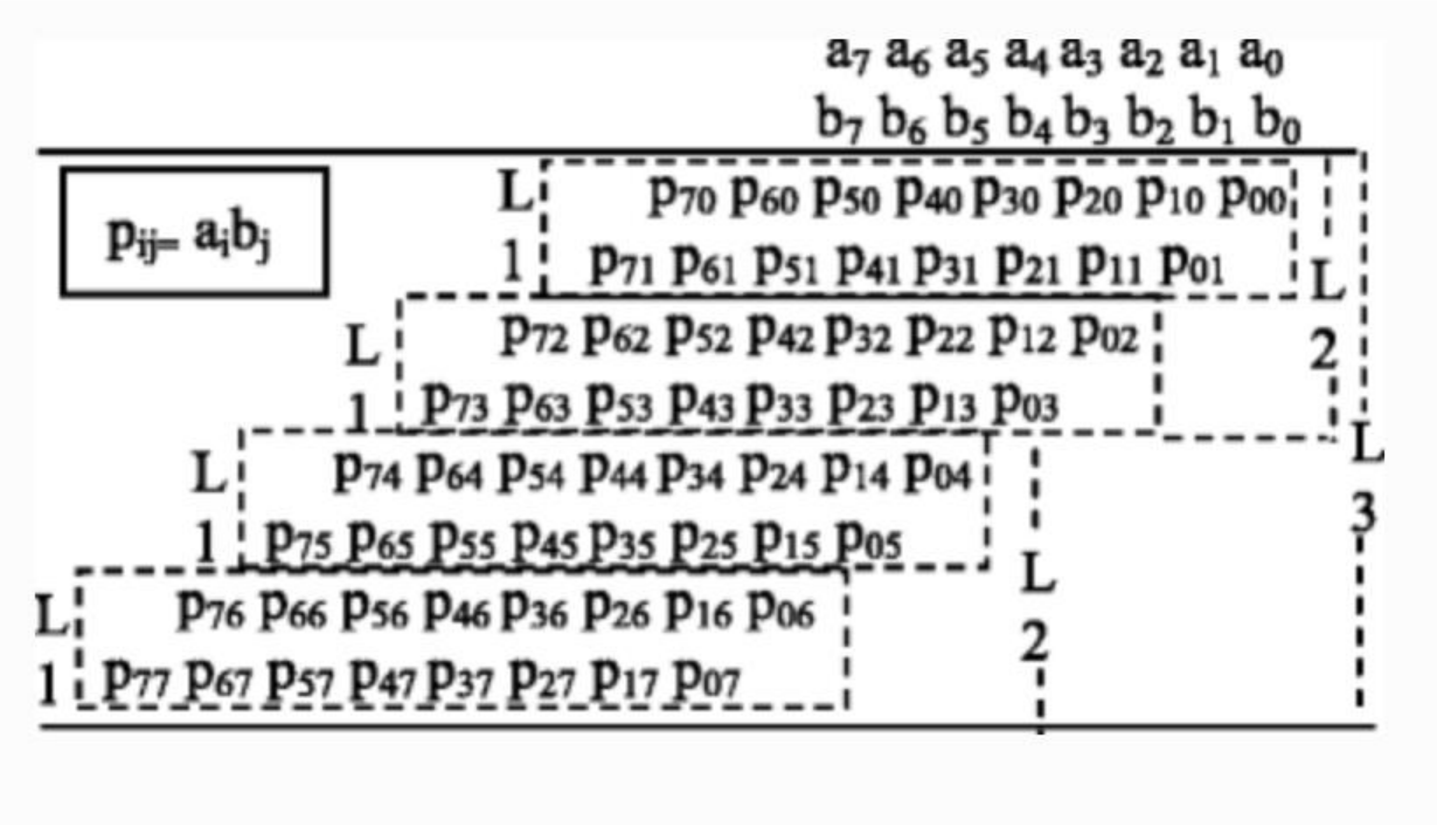
- ❑ Multiplier
- ❑ Pipeline
- ❑ **Fast Multiplier**
- ❑ Assignment
- ❑ Appendix

Popular Techniques

- ❑ Carry-save adder
- ❑ Wallace tree
- ❑ High-radix
- ❑ And more …

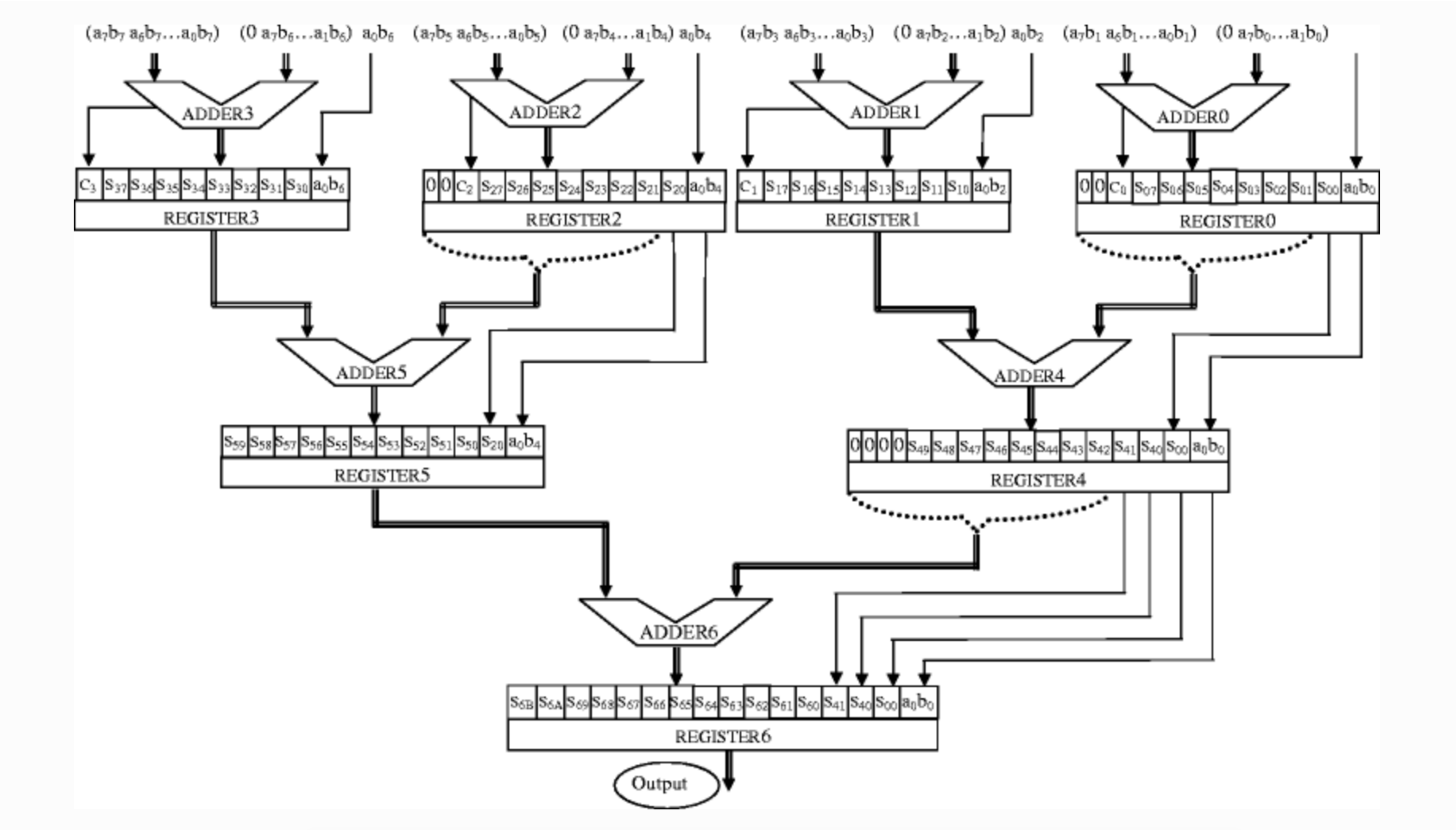
https://www.slideshare.net/siddiqueibrahim37/pipelining-41608675

A Hierarchical Design


a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0
b_7 b_6 b_5 b_4 b_3 b_2 b_1 b_0
$p_{ij}= a_ib_j$
L 1
p_{70} p_{60} p_{50} p_{40} p_{30} p_{20} p_{10} p_{00}
p_{71} p_{61} p_{51} p_{41} p_{31} p_{21} p_{11} p_{01}
L 2
L 1
p_{72} p_{62} p_{52} p_{42} p_{32} p_{22} p_{12} p_{02}
p_{73} p_{63} p_{53} p_{43} p_{33} p_{23} p_{13} p_{03}
L 3
L 1
p_{74} p_{64} p_{54} p_{44} p_{34} p_{24} p_{14} p_{04}
p_{75} p_{65} p_{55} p_{45} p_{35} p_{25} p_{15} p_{05}
L 2
L 1
p_{76} p_{66} p_{56} p_{46} p_{36} p_{26} p_{16} p_{06}
p_{77} p_{67} p_{57} p_{47} p_{37} p_{27} p_{17} p_{07}

https://link.springer.com/article/10.1007/s11265-012-0657-7

A Hierarchical Design


$(a_7b_7\ a_6b_7 \ldots a_0b_7)$
$(0\ a_7b_6 \ldots a_1b_6)$
a_0b_6
$(a_7b_5\ a_6b_5 \ldots a_0b_5)$
$(0\ a_7b_4 \ldots a_1b_4)$
a_0b_4
$(a_7b_3\ a_6b_3 \ldots a_0b_3)$
$(0\ a_7b_2 \ldots a_1b_2)$
a_0b_2
$(a_7b_1\ a_6b_1 \ldots a_0b_1)$
$(0\ a_7b_0 \ldots a_1b_0)$
ADDER3
ADDER2
ADDER1
ADDER0
C_3 S_{37} S_{36} S_{35} S_{34} S_{33} S_{32} S_{31} S_{30} a_0b_6
REGISTER3
0 0 C_2 S_{27} S_{26} S_{25} S_{24} S_{23} S_{22} S_{21} S_{20} a_0b_4
REGISTER2
C_1 S_{17} S_{16} S_{15} S_{14} S_{13} S_{12} S_{11} S_{10} a_0b_2
REGISTER1
0 0 C_0 S_{07} S_{06} S_{05} S_{04} S_{03} S_{02} S_{01} S_{00} a_0b_0
REGISTER0
ADDER5
ADDER4
S_{59} S_{58} S_{57} S_{56} S_{55} S_{54} S_{53} S_{52} S_{51} S_{50} S_{20} a_0b_4
REGISTER5
0 0 0 0 S_{49} S_{48} S_{47} S_{46} S_{45} S_{44} S_{43} S_{42} S_{41} S_{40} S_{00} a_0b_0
REGISTER4
ADDER6
S_{6B} S_{6A} S_{69} S_{68} S_{67} S_{66} S_{65} S_{64} S_{63} S_{62} S_{61} S_{60} S_{41} S_{40} S_{00} a_0b_0
REGISTER6
Output

https://link.springer.com/article/10.1007/s11265-012-0657-7

Outline

- ❑ Multiplier
- ❑ Pipeline
- ❑ Fast Multiplier
- ❑ **Assignment**
- ❑ Appendix

Execution

- Download the sample code
- Compile
  - `> iverilog -o lab2.vvp lab2.v`
- Simulate
  - `> vvp lab2.vvp`
    - Generates `lab2.vcd`
- View waveform
  - `> gtkwave lab2.vcd lab2.sav`

Requirements (1/2)

- ❑ Replace **`<index>`** in **`lab2.v`** with proper indexes
  - ➢ Leave the rest of the code unchanged
- ❑ Show your source code
- ❑ Show the waveform with the settings in **`lab2.sav`**
- ❑ What is the latency?
  - ➢ Worst case waiting time from the input becomes steady to the register of the last stage refreshes
  - ➢ Use tick as time unit

Requirements (2/2)

- Minimize the clock cycle by **changing the delays** in the module **`mult_tb`**
  - What is the minimum clock cycle?
    - Use tick as time unit
  - Show the waveform with the settings in **`lab2.sav`**
    - Should include 1750 to 1800 sec

Outline

- □ Multiplier
- □ Pipeline
- □ Fast Multiplier
- □ Assignment
- □ **Appendix**

Multiplier and Pipelining

- Signed multiplier
  - http://courses.csail.mit.edu/6.111/f2008/handouts/L09.pdf
- Details on pipelining
  - https://www.slideshare.net/siddiqueibrahim37/pipelining-41608675
- Details on fast multiplier
  - https://link.springer.com/article/10.1007/s11265-012-0657-7

Some Details on Verilog

- Expression bit width
  - http://yangchangwoo.com/podongii_X2/html/technote/TOOL/MANUAL/21i_doc/data/fndtn/ver/ver4_4.htm
- Event queue
  - Determine the order of different types of assignment
  - http://www.ece.lsu.edu/v/2015/lsli-event-q.pdf
- Transport delay and inertial delay
  - http://www-inst.eecs.berkeley.edu/~cs152/fa06/handouts/CummingsHDLCON1999_BehavioralDelays_Rev1_1.pdf

Q&A