SNS COLLEGE OF ENGINEERING

(Autonomous)

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

Computer Organization and Architecture

Prepared by,

K.Sangeetha

Assistant Professor/ECE

SNS College of Engineering

Faster Multplication

Fast Multiplication

There are two techniques discussed for speeding the Multiplication Operation

i. Reducing Maximum number of Summands: - Bit Pair Recoding of Multipliers reduces the maximum number of summands (versions of multiplicand) that must be added to n/2 for n bit operands[1].

ii. Faster Summands addition: - Summand addition uses

   a) Carry save – In this carry generated by Full Adders in i^{th} row propagated to $(i+1)^{th}$ row Full Adders instead of rippling through adder in same row and

   b) Summands Reduction Technique - Techniques like 3 to 2, 4 to 2 reduction of summands [1].

Faster Multplication

i. **Reducing Maximum number of Summands using Bit Pair Recoding of Multipliers**

Bit-pair recoding of the multiplier – It is a modified Booth Algorithm, In this it uses one summand for each pair of booth recoded bits of the multiplier.

Step 1: Convert the given Multiplier into a Booth Recode the Multiplier.
Step 2: Group the recoded Multiplier bits in pairs and observe the following

Booth Multiplication

Table 3.3: Table of multiplicand selection decisions

Normal Multiplier			Booth Algorithm		Modified Booth Algorithm	
Multiplier bit -pair		Multiplier bit on the Right	Booth recoded multiplier		Bit pair recoded Value Multiplicand Selected at Position i	
i+1	i	i-1	i+1	i	i	i
0	0	0	0	0	$(0x2^1 - 0x2^0)M$	0xM
0	0	1	0	+1	$(0x2^1 + 1x2^0)M$	+1xM
0	1	0	+1	-1	$(+1x2^1 - 1x2^0)M$	+1xM
0	1	1	+1	0	$(+1x2^1 - 1x2^0)M$	+2xM
1	0	0	-1	0	$(-1x2^1 - 0x2^0)M$	-2xM
1	0	1	+1	-1	$(+1x2^1 - 1x2^0)M$	+1xM
1	1	0	0	-1	$(0x2^1 - 1x2^0)M$	-1xM
1	1	1	0	0	$(0x2^1 - 0x2^0)M$	0xM

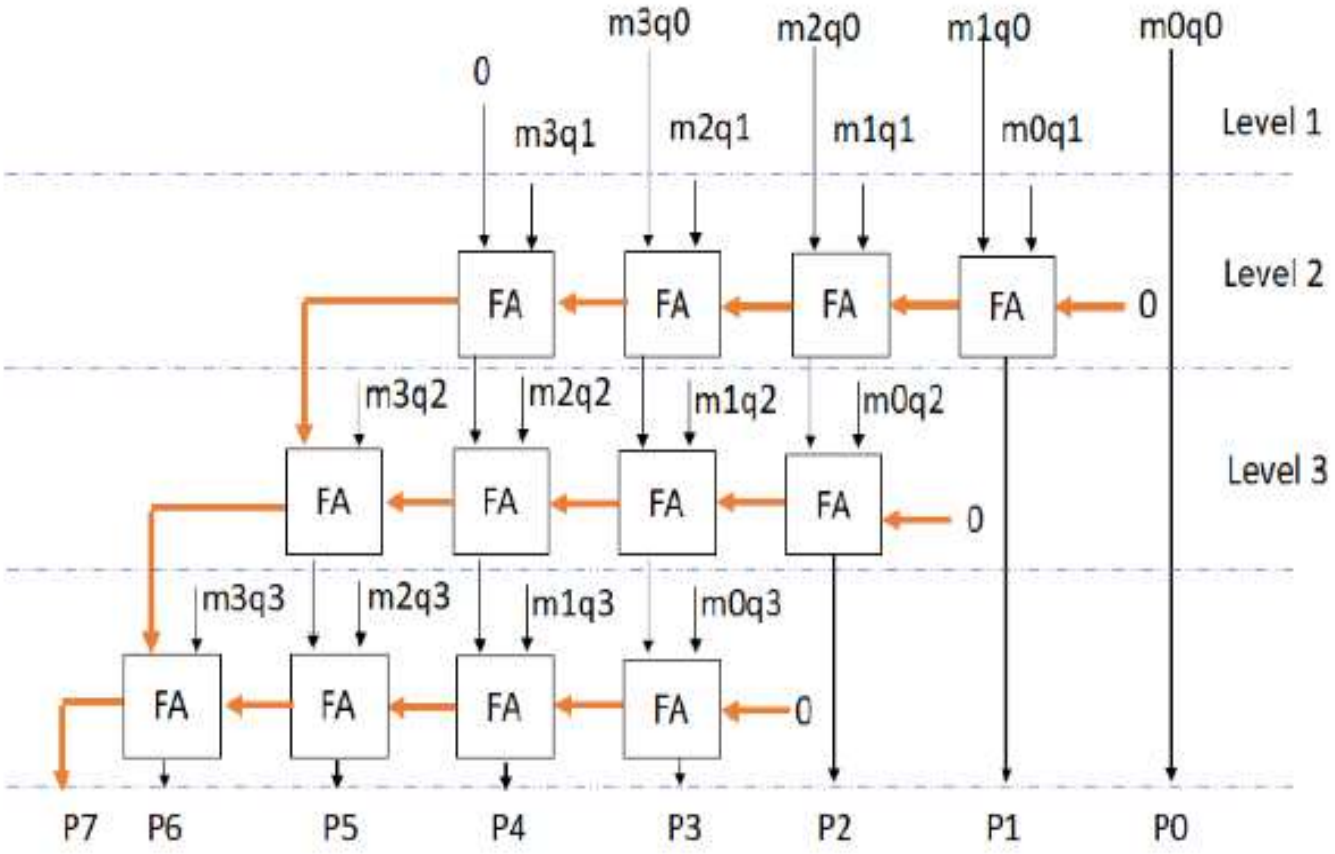
Carry – Save Addition

ii. Faster Summands addition

a) Carry Save Addition of Summands

- As we know Multiplication involves the addition of several summands. A technique called carry- save addition (CSA) can be used to speed up the process.
- Let us consider the 4 × 4 multiplication array (m3 m2 m1 m0 multiplied with q3 q2 q1 q0)
- Use of Full Adders in first row is not needed as there is No addition of summands involved, therefore in first row (Level1) is made to consists of just the AND gates that produce the partial products (m3q0, m2q0, m1q0 and m0q0). From second row onwards n-bit full adders are use and it can be seen that in Second row each full adder takes summands of first row as one input and summands of second row as second input with carry rippling from in row. This basically reduces the number of Full Adder by n.

Carry – Save Addition cont..



4x4 Multiplication using Ripple Carry Array

Carry – Save Addition cont..

- In this, Even though the number of Full Adder are reduced by n numbers, It can be still seen that the Carry is getting rippled from an adder to the other adder in each row (of same level) which basically delays the summands addition.

- Instead of letting the carries ripple along the rows (ith row), they can be “saved” and introduced into the next row i.e., (i+1)th row, at the correct weighted positions, as shown in Figure 3.26 Carry Save Addition.

- As carry in ith row propagates to (i+1)th row, it frees up Cin input of three full adders in the Second row (Full adder at LSB takes carry input as Zero). Now these inputs are now used to introduce the third summand bits m2q2, m1q2, and m0q2. The summand m3q2 goes as input to the FA at the left end of next successive row [1]

Carry – Save Addition cont..

The Carry Save Addition of summands for n=m=4 is as shown below.

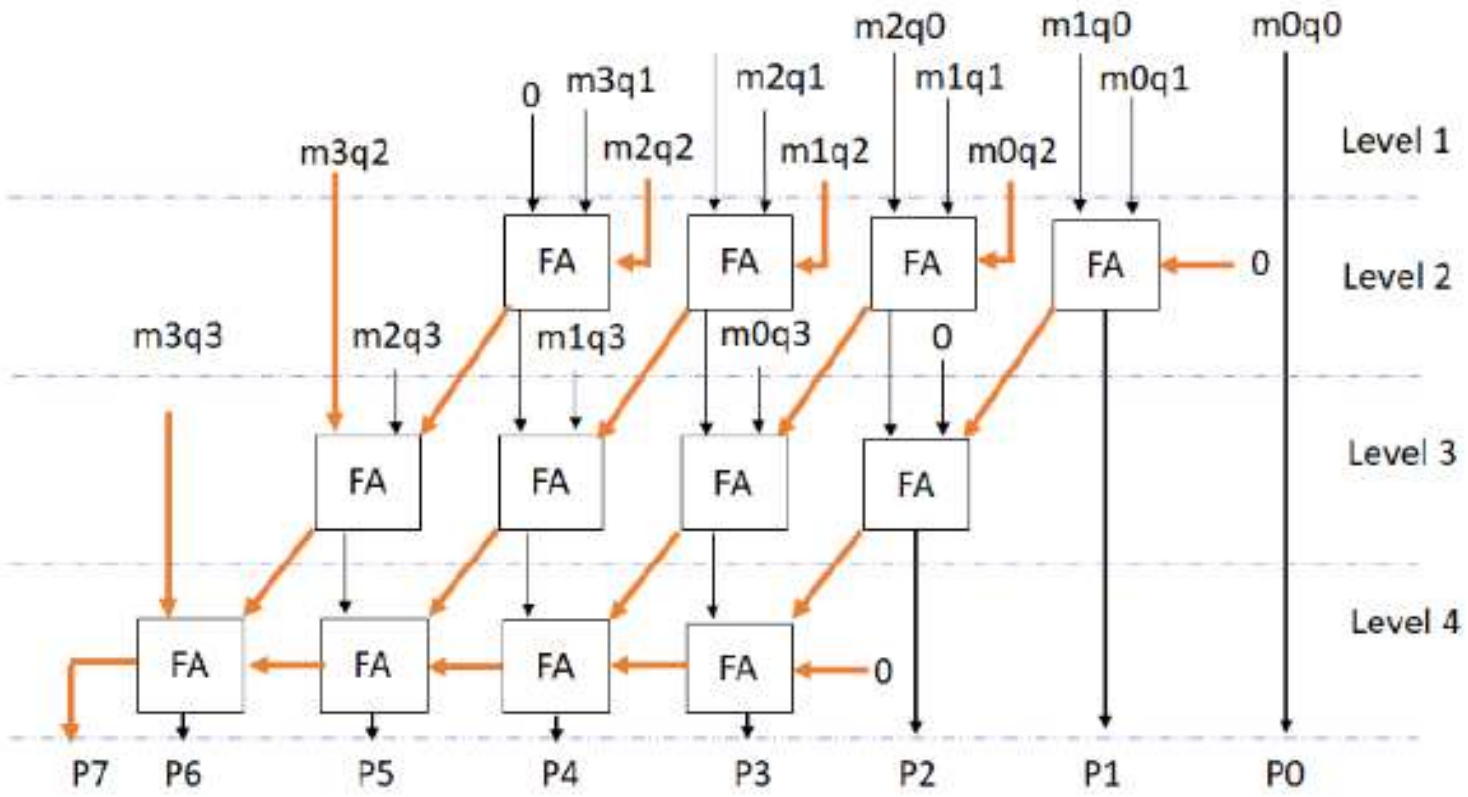


Figure 3.26: 4x4 Multiplication using Carry Save Array

Carry – Save Addition cont..

Now, two inputs of each of three full adders in the third row (level 3) are fed by the sum and carry outputs from the Second row (Level 2). The third input is used to introduce the bits m2q3, m1q3, and m0q3 of the fourth summand. The high-order bit m3q3 goes input to the Full Adder at the left end of next successive row. The saved carry bits and the sum bits from the third row are now added in the fourth row (Level 4), which is a ripple-carry adder, to produce the final product bits [1].

The delay through the carry-save array is somewhat less than the delay through the ripple-carry array. This is because the S and C vector outputs from each row are produced in parallel in one full-adder delay [1]

Assessment

1. Difference between Computer Architecture and Organization.
2. List the three types of Busses.
3. What are the different types of Memory.
4. Define ISA.
4. Define DataPath.


thank
you