Chapter 03: Computer Arithmetic

Lesson 07:

Integer Division

Objective

- Understand process of integer division
- Restoring Algorithm
- Non-restoring Algorithm

Division using successive subtraction

Division using successive subtraction

- Implemented on computer systems by repeatedly subtracting the divisor from the dividend
- Counting the number of times that the divisor can be subtracted from the dividend before the dividend becomes smaller than the divisor

Division 15 with 5

- Subtract repeatedly from 15, getting 10, 5, and 0 as intermediate results
- The quotient, 3, is the number of subtractions that had to be performed before the intermediate result became less than the dividend

15 ÷ 5

```
           0b0101
       ___________
0b 11 ) 0b1111
          00
        ______
          1111
          11
        ______
          0011
           00
        ______
            11
            11
        ______
          0b00
        ______
```

Too long Time

- For example, 2^{31} (one of the larger numbers representable in 32-bit unsigned integers) divided by 2 is 2^{30}, meaning that 2^{30} subtractions would have to be done to perform this division by repeated subtraction
- On a system operating at 1 GHz, this would take approximately 1 s, far longer than any other arithmetic operation

Division using look-up table

Lookup Table Method

- Using pre-generated tables, these techniques generate 2 to 4 bits of the quotient in each cycle
- This allows 32-bit or 64-bit integer divisions to be done in a reasonable number of cycles

Division using Restoring Algorithm

Restoring Algorithm

- Assume— X register *k*-bit dividend
- Assume— Y the *k*-bit divisor
- Assume – S a sign-bit

Restoring Algorithm

1. Start: Load 0 into accumulator k-bit A and dividend X is loaded into the k-bit quotient register MQ.
2. Step A: Shift $2k$-bit register pair A-MQ left
3. Step B: Subtract the divisor Y from A.

Restoring Algorithm

4. Step C: If sign of A (msb) = 1, then reset MQ_0 (lsb) = 0 else set = 1.
5. Steps D: If MQ_0 = 0 add Y (restore the effect of earlier subtraction).
6. Steps A to D repeat again till the total number of cyclic operations = k.

At the end, *A* has the remainder and *MQ* has the quotient

Division of 4-bit number by 7-bit dividend

Step	S-flag *	First Register for A	Second Register for MQ	Action Taken	Number of operations (instructions)
Start	0	0b 0000	0b 0000	Clear S, A, MQ	3 for clearing C, A and M
	0	0b 0001	0b 1110	Load dividend X (lower k bits) between MQ_{k-1} and MQ_0 and dividend higher bits in A	2 for loading A and MQ
Step 0A	0	0011	1100	Shift left S-A-M	2
Step 0B	0	0000	1100	Subtract Y from S-A, result in S-A	1
Step 0C	0	0000	1101	$MQ_0 = 1$ as S = 0	1
Step 0D	0	0000	1101	Skip restore by adding as S = 0	1 (test S)
Step 1A	0	0001	1010	Shift left S-A-M	2
Step 1B	1	1110	1010	Subtract Y from S-A, result in S-A	1
Step 1C	1	1110	1010	$MQ_0 = 0$ as S = 1	1
Step 1D	0	0001	1010	Add Y into S-A to restore as S = 1	1

Division of 4-bit number by 7-bit dividend

Step 2A	0	0011	0100	Shift left S-A-M	2
Step 2B	0	0000	0100	Subtract *Y* from S-*A*, result in S-*A*	1
Step 2C	0	0000	0101	$MQ_0 = 1$ as S = 0	1
Step 2D	0	0000	0101	Skip restore as S = 0	1(test S)
Step 3A	0	0000	1010	Shift left S-A-M	2
Step 3B	1	1101	1010	Subtract *Y* from S-*A*, result in S-*A*	1
Step 3C	1	1101	1010	$MQ_0 = 0$ as S = 1	1
Step 3D	0	0000	1010	Add Y into S-A to restore as S = 1	1
Answer	0	Remainder = 0,		Quotient Decimal 10	Total 25

* after the left shift from *msb* of *A*.

Division using Non-restoring Algorithm

Non-Restoring Algorithm

- Assume— that there is an accumulator and MQ register, each of *k*-bits
- MQ_0, (lsb of MQ) bit gives the quotient, which is saved after a subtraction or addition

Non-Restoring Algorithm

- Total number of additions or subtractions are k-only and total number of shifts = k plus one addition for restoring remainder if needed

Non-Restoring Algorithm

- Assume— that X register has $(2k–1)$ bit for dividend and Y has the k-bit divisor
- Assume— a sign-bit S shows the sign

Non- Restoring Algorithm

1. Load (upper half *k–1* bits of the dividend X) into accumulator *k*-bit A and load dividend X (lower half bits into the lower *k* bits at quotient register MQ

- Reset sign $S = 0$
- Subtract the *k* bits divisor Y from S-A (1 plus *k* bits) and assign MQ_0 as per S

Non- Restoring Algorithm

2. If sign of A, $S = 0$, shift S plus 2*k*-bit register pair A-MQ left and subtract the *k* bits divisor Y from *S*-A (1 plus *k* bits); *else if* sign of A, $S = 1$, shift S plus 2*k*-bit register pair *A-MQ* left and add the divisor Y into *S*-A (1 plus *k* bits)

- Assign MQ_0 as per S

Non- Restoring Algorithm

3. Repeat step 2 again till the total number of operations = *k*.
4. If at the last step, the sign of *A* in *S* = 1, then add Y into *S-A* to leave the correct remainder into *A and* also assign MQ_0 as per S, else do nothing.
5. *A* has the remainder and *MQ* has the quotient

Division of 4-bit number by 7-bit dividend by Non Restoring Algorithm

Step	S-flag *	First Register for A	Second Register for MQ	Action Taken	Number of operations (instructions)
Start	0	0b0000	0b0000	Clear S, A, MQ	3 for clearing C, A and M
	0	0b0001	0b1110	Load dividend X (lower k bits) in MQ_{k-1} and MQ_0 and dividend higher k–1 bits in A	2 for loading A and MQ
Step 0A	1	1110	1110	Subtract Y from S-A, because S = 0 result in S-A	1
Step 0B	1	1110	1110	MQ_0 = 0 as S = 1	1
Step 0C	1	1101	1100	Shift left S-A-M	2

Division of 4-bit number by 7-bit dividend by Non Restoring Algorithm

Step 1A	0	0000	1100	Add Y into S-A, because S = 1	1
Step 1B	0	0000	1101	MQ_0 = 1 as S = 0	1
Step 1C	0	0001	1010	Shift left S-A-M	2
Step 2A	1	1110	1010	Subtract Y into S-A, because S = 0	1
Step 2B	1	1110	1010	MQ_0 = 0 as S = 1	1
Step 2C	1	1101	0100	Shift left S-A-M	2
Step 3A	1	0000	0100	Add Y into S-A, because S = 1	1
Step 3B	0	0000	0101	MQ_0 = 1 as S = 0	1
Step 3C	0	0000	1010	Shift C-A-M	2
Last	0	0000	1010	Do not Add Y into S-A, because S = 0 and make no change in MQ_0	1
Answer	0	Remainder = 0,		Quotient Decimal 10	Total 22

Summary

We learnt

- Division by successive subtraction is slowest
- Restoring Algorithm
- Non-Restoring Algorithm

End of Lesson 07 on

Integer Division