CSARCH Lecture Series: Non-Restoring Division

Sensei RL Uy
College of Computer Studies
De La Salle University
Manila, Philippines

'2110

Copyright Notice

Overview

Reflect on the following question:

- How does Arithmetic and Logic Unit (ALU) perform division?

```
int main()
{
    int var, var1, var2;
    var = 5;
    var1 = 2;
    var2 = var / var1;
}
```

Overview

- This sub-module describes how Arithmetic Logic Unit (ALU) performs division using non-restoring division
- The objective is as follows:
  - ✓Describe the process of performing non-restoring division

Integer Division

- Unlike subtraction, division cannot easily be performed using multiplication since it is difficult to take the reciprocal of a number.
- Division is performed by repeated subtraction
- There are two methods for integer binary division
  - Restoring
  - Non-restoring

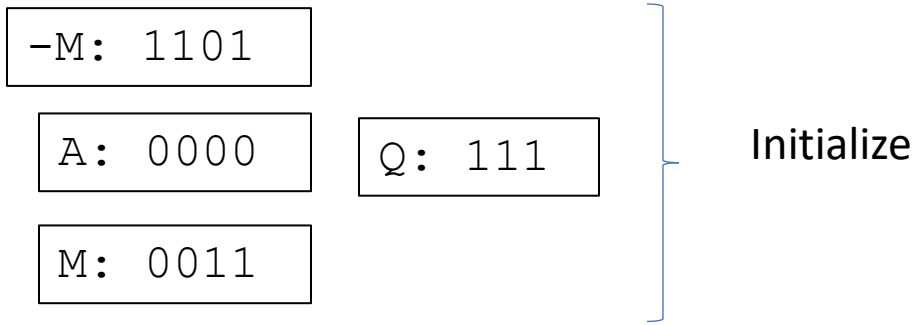
Non-restoring Method

- The non-restoring division defers restoration to reduce operations
  - Addition is performed instead of subtraction in the next step
  - This requires a final restoration, when last operation yielded a negative remainder

Non-Restoring Method

- Initialization
  - Clear *A*. Requires 1 extra bit for *A* to be used as a sign bit.
  - *Q* gets dividend.
  - *M* gets divisor.
- Loop for each bit of the dividend *Q*
  - If *A* (previous result) is negative ($A_n = 1$)
    - Shift *AQ* to the left.
    - Add ($A \leftarrow A + M$)
    - Q_0 gets the complement value of the MSB of *A*
  - Else
    - Shift *AQ* to the left.
    - Subtract ($A \leftarrow A - M$)
    - Q_0 gets the complement value of the MSB of *A*
- Perform restoration if last result in *A* is negative, *i.e.*, if $A_n = 1$ then $A \leftarrow A + M$
- Quotient in *Q* while remainder in *A*; adjust sign as needed

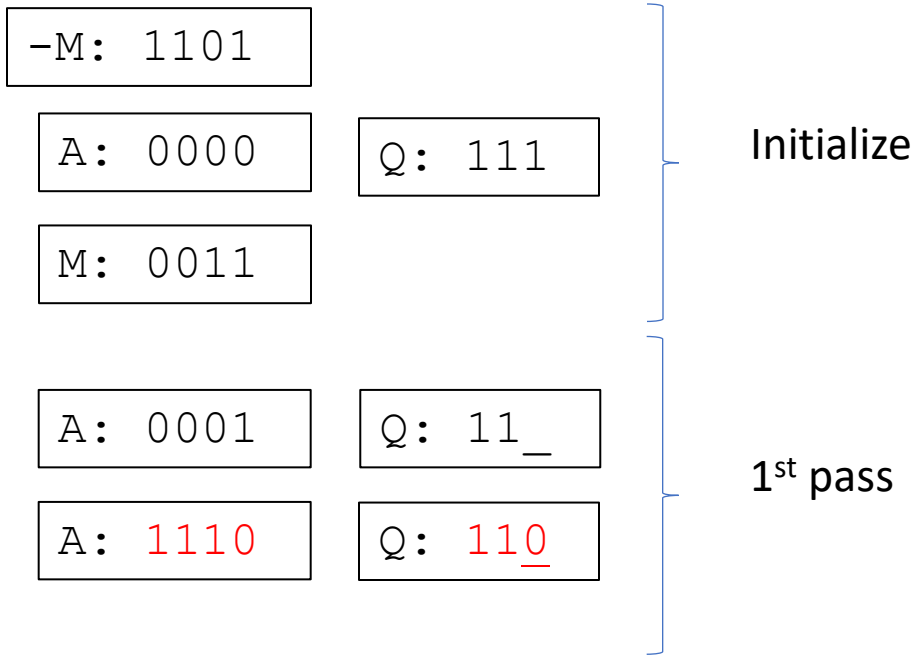
Non-Restoring Division



111 (Q) / 11 (M)

- Initialization
  - Clear *A*. Requires 1 extra bit for *A* to be used as a sign bit.
  - *Q* gets dividend.
  - *M* gets divisor.
- Loop for each bit of the dividend *Q*
  - If *A* (previous result) is negative ($A_n = 1$)
    - Shift *AQ* to the left.
    - Add ($A \leftarrow A + M$)
    - Q_0 gets the complement value of the MSB of *A*
  - Else
    - Shift *AQ* to the left.
    - Subtract ($A \leftarrow A - M$)
    - Q_0 gets the complement value of the MSB of *A*
- Perform restoration if last result in *A* is negative, *i.e.*, if $A_n = 1$ then $A \leftarrow A + M$
- Quotient in *Q* while remainder in *A*; adjust sign as needed

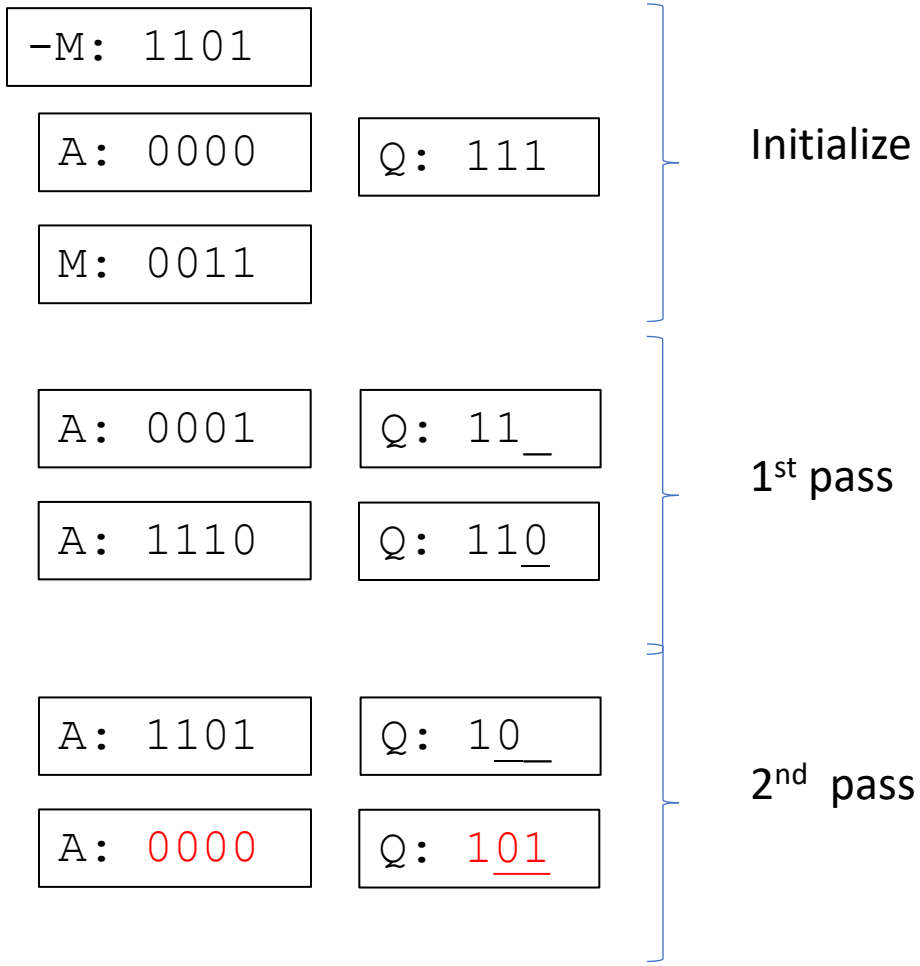
Non-Restoring Division



111 (Q) / 11 (M)

- Initialization
  - Clear *A*. Requires 1 extra bit for *A* to be used as a sign bit.
  - *Q* gets dividend.
  - *M* gets divisor.
- Loop for each bit of the dividend *Q*
  - If *A* (previous result) is negative ($A_n = 1$)
    - Shift *AQ* to the left.
    - Add ($A \leftarrow A + M$)
    - Q_0 gets the complement value of the MSB of *A*
  - Else
    - Shift *AQ* to the left.
    - Subtract ($A \leftarrow A - M$)
    - Q_0 gets the complement value of the MSB of *A*
- Perform restoration if last result in *A* is negative, *i.e.*, if $A_n = 1$ then $A \leftarrow A + M$
- Quotient in *Q* while remainder in *A*; adjust sign as needed

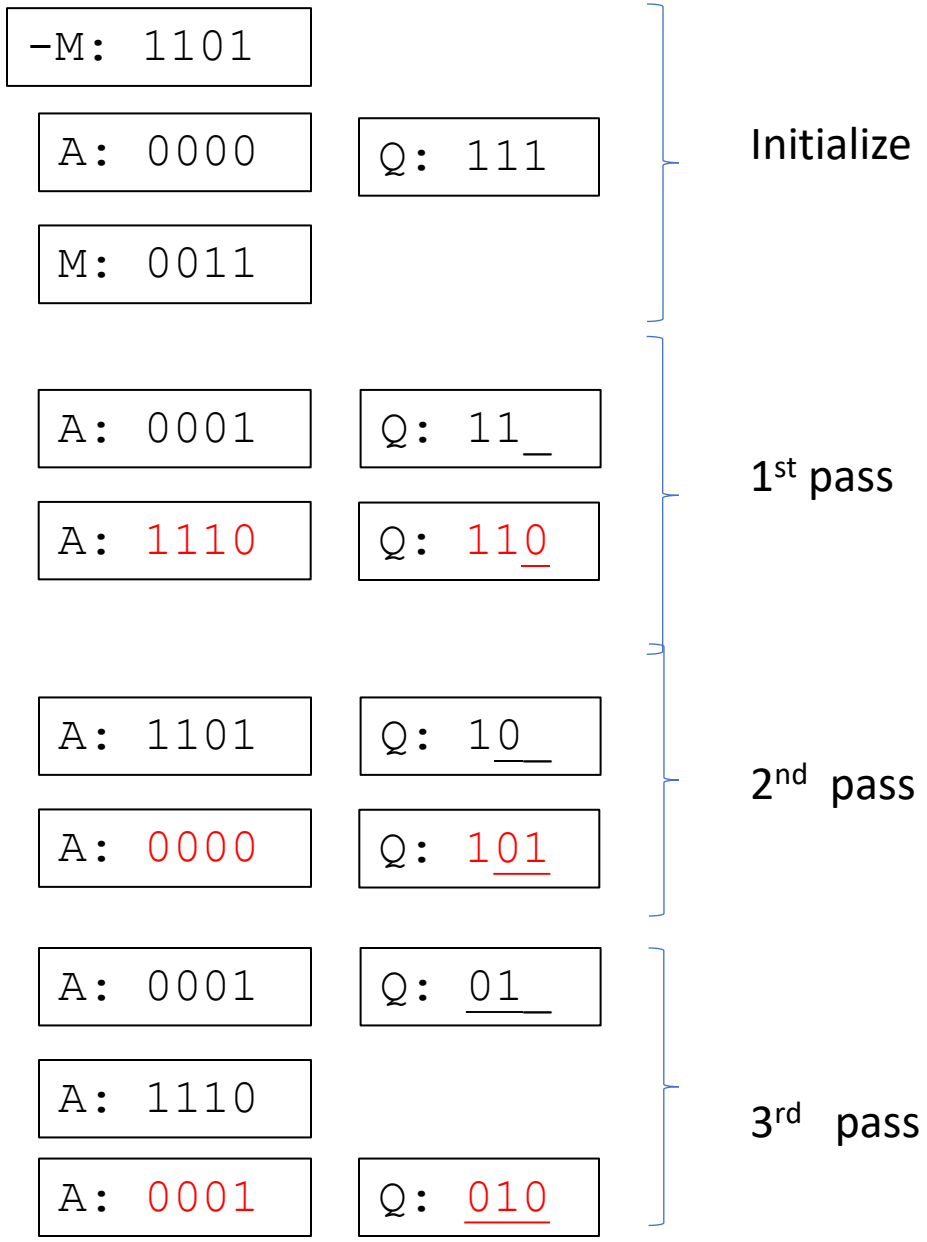
Non-Restoring Division



111 (Q) / 11 (M)

- Initialization
  - Clear *A*. Requires 1 extra bit for *A* to be used as a sign bit.
  - *Q* gets dividend.
  - *M* gets divisor.
- Loop for each bit of the dividend *Q*
  - If *A* (previous result) is negative ($A_n = 1$)
    - Shift *AQ* to the left.
    - Add ($A \leftarrow A + M$)
    - Q_0 gets the complement value of the MSB of *A*
  - Else
    - Shift *AQ* to the left.
    - Subtract ($A \leftarrow A - M$)
    - Q_0 gets the complement value of the MSB of *A*
- Perform restoration if last result in *A* is negative, *i.e.*, if $A_n = 1$ then $A \leftarrow A + M$
- Quotient in *Q* while remainder in *A*; adjust sign as needed

Non-Restoring Division



111 (Q) / 11 (M)

- Initialization
  - Clear *A*. Requires 1 extra bit for *A* to be used as a sign bit.
  - *Q* gets dividend.
  - *M* gets divisor.
- Loop for each bit of the dividend *Q*
  - If *A* (previous result) is negative ($A_n = 1$)
    - Shift *AQ* to the left.
    - Add ($A \leftarrow A + M$)
    - Q_0 gets the complement value of the MSB of *A*
  - Else
    - Shift *AQ* to the left.
    - Subtract ($A \leftarrow A - M$)
    - Q_0 gets the complement value of the MSB of *A*
- Perform restoration if last result in *A* is negative, *i.e.*, if $A_n = 1$ then $A \leftarrow A + M$
- Quotient in *Q* while remainder in *A*; adjust sign as needed

Try

Try: 01101 (Q) / 00101 (M) (using non-restoring division)
Show the value of A and Q after the end of each pass

After this pass	A	Q
1st		
2nd		
3rd		
4th		
5th		

Try

Try: 01101 (Q) / 00101 (M) (using non-restoring division)
Show the value of A and Q after the end of each pass

After this pass	A	Q
1st	111011	11010
2nd	111100	10100
3rd	11110	01000
4th	000001	10001
5th	000011	00010

To recall …

- What have we learned:
  - ✓Describe the process of performing non-restoring division