はじめに

- 今回のテーマは、除算アルゴリズムと除算器について講義します。

- 教科書の対応範囲は、以下の通りです。
  - 6.1.4項(a)～(e)（p.189～199）

- さて、これから除算器の設計について考えていきますが、大まかに、以下の2つのアプローチがあります。
  - ハードウェアアルゴリズム
    - 除算器を、すべてハードウェア（論理回路）で実現します。
    - ハードウェア向きの解法を用いて除算を行います。
      - 引き戻し法や、引き放し法、などがあります。
  - ソフトウェア的解法
    - 除算機能のすべてをハードウェアで実現するのではなく、マイクロプログラム等で制御します。
      - アルゴリズムとしては、ニュートン法などが使われます。

ソフトウェア的解法

- 本講義では、ニュートン法を用いた除算アルゴリズムについて解説します。
  - ニュートン法は、一般に、方程式を解くためのアルゴリズムとして知られています。
    - 皆さんの中には、ニュートン法を用いて方程式を解くプログラムを書いたことがある人がいるかもしれませんね。
- $X \div Y = X \times (1/Y)$ですから、$Y$の逆数が求まれば、あとは$X$と掛けるだけです。そこで、逆数を求めるのにニュートン法を使用します。
  - 加減算と乗算を用いて除算を行いますので、速い浮動小数点乗算器を装備していることが前提です。

ニュートン法による逆数の求め方

- aの逆数は、以下の方程式の解です。

$$\frac{1}{a} = x \Leftrightarrow \frac{1}{x} - a = 0$$

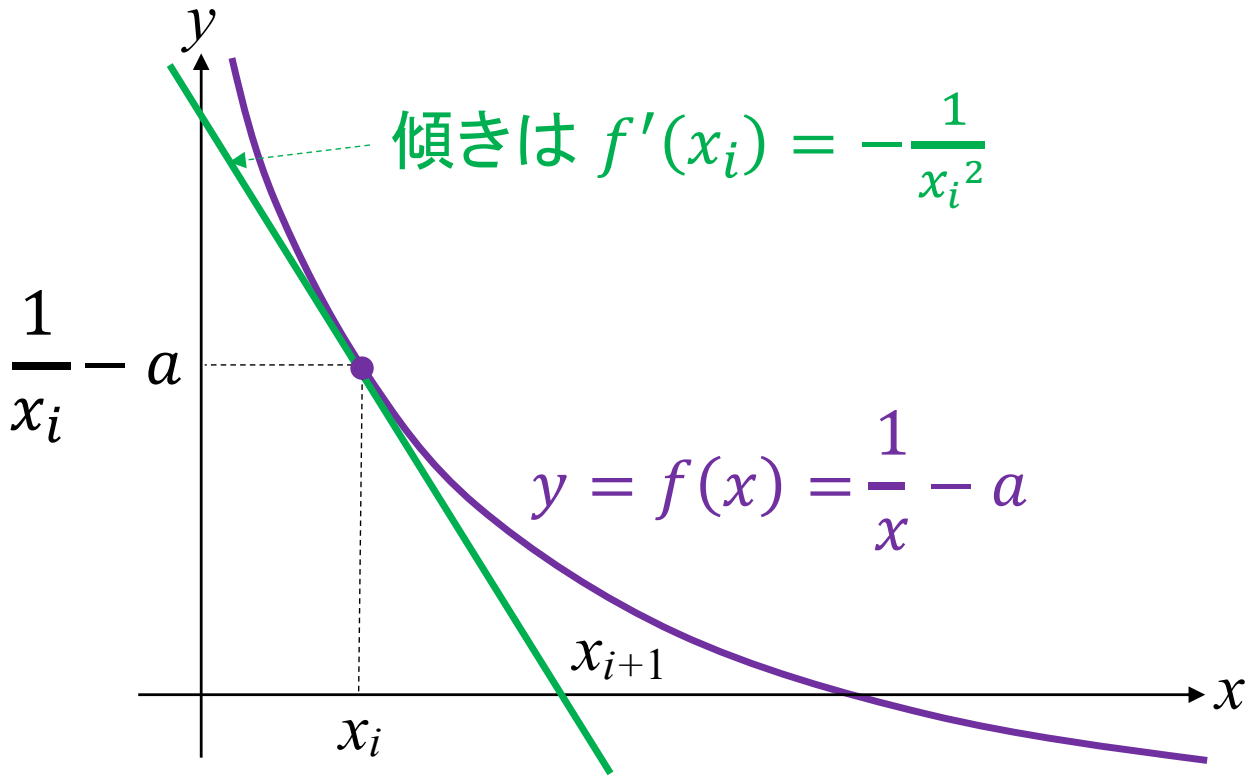
- そこで、以下のように関数 $f(x)$を定義します。

$$f(x) = \frac{1}{x} - a$$

- このとき、

$$\frac{df(x)}{dx} = f'(x) = -\frac{1}{x^2}$$

- $y=f(x)$ の $x=x_i$ における接線を考えます。



- 接線の方程式は、

$$y-\left(\frac{1}{x_i}-a\right)=-\frac{1}{{x_i}^2}(x-x_i)$$

- この接線がx軸と交わる点を$(x_{i+1}, 0)$とすると、

$$0 - \left(\frac{1}{x_i} - a\right) = -\frac{1}{{x_i}^2}(x_{i+1} - x_i)$$

- これより、以下の漸化式が得られます。

$$x_{i+1} = 2x_i - a{x_i}^2 = (2 - ax_i)x_i$$

  - 乗算と減算だけで、x_iからx_{i+1}が求まります。
  - この計算を繰り返せば、$\{x_i\}$は次第に$f(x)=0$の解に近づきます。
    - 3回程度の繰り返しで十分な精度（桁数）の解が得られるように、$\{x_i\}$の初期値のテーブルを用意しておきます。

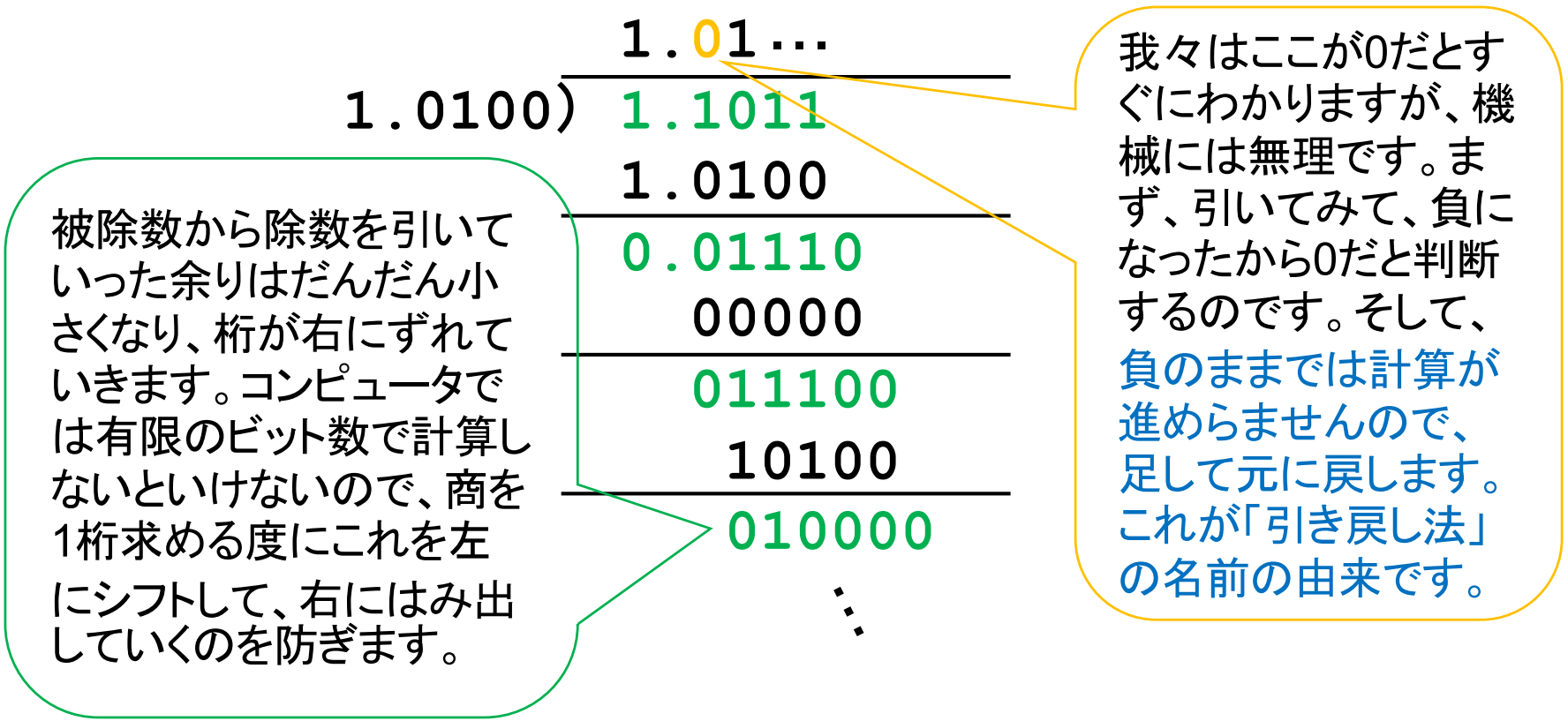
ハードウェアアルゴリズム

- ハードウェア（論理回路）での処理向きに、加減算とシフト演算のみを用いて除算を行います。
- 配列型の除算器も考えられますが、あまりメリットがないので実際には使用しません。
- よって、本講義では、繰り返し型の除算器を前提に、以下の除算アルゴリズムを解説します。
  - 引き戻し法（restoring division）
  - 引き放し法（nonrestoring division）

引き戻し法

- 整数でやっても同じことなのですが、僅かながらも簡単になるので、ここでは正規化された浮動小数点数の仮数部（つまり、「1.～」の形の正の数）を対象に、割り算の計算方法を考えることにします。
  - なお、整数の場合は余りについてもちゃんと考えなければなりません。
  - 浮動小数点数の場合は、余りを求めることはありませんが、丸めるために、余分の桁数まで含めて商を求める必要があります。

- 計算原理としては、我々が筆算で計算するのと全く同じです。
  - 例えば、1.1011÷1.0100 の計算は、

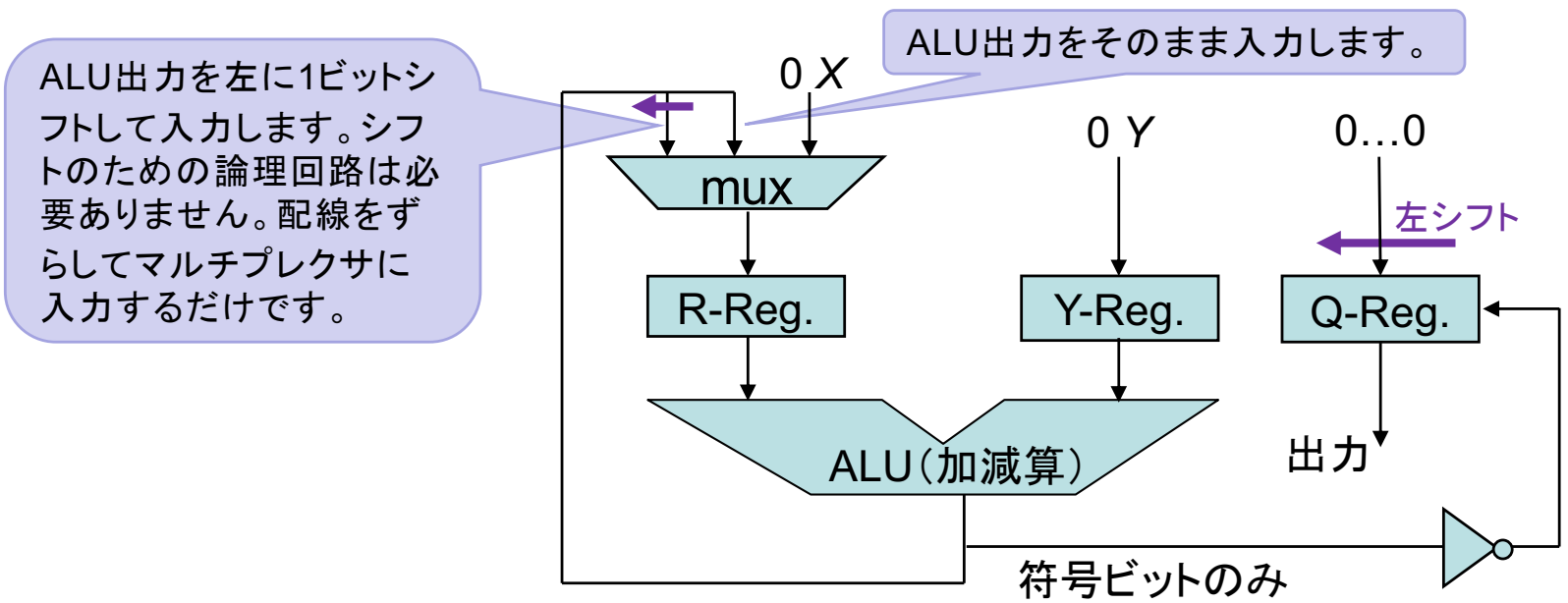


– コンピュータで計算する場合は、

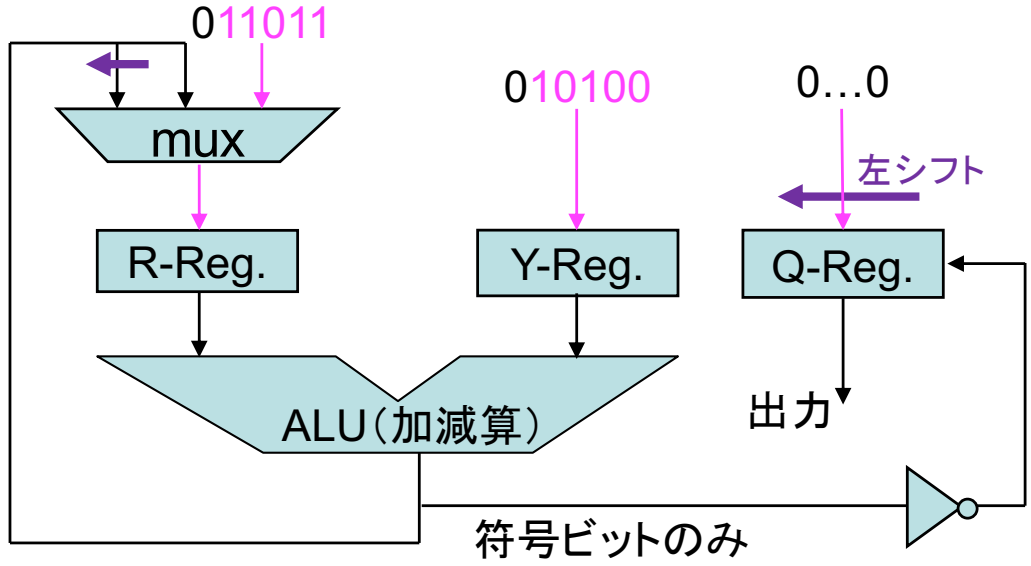
符号ビットを追加して計算します。

```
    01.1011
–)  01.0100
-------------
    00.0111   正なので Q=1.
    00.1110   左にシフト
–)  01.0100
-------------
    11.1010   負なので Q=1.0
+)  01.0100   （足して元に戻す）
-------------
    00.1110
    01.1100   左にシフト
–)  01.0100
-------------
    00.1000   正なので Q=1.01
      ⋮
```

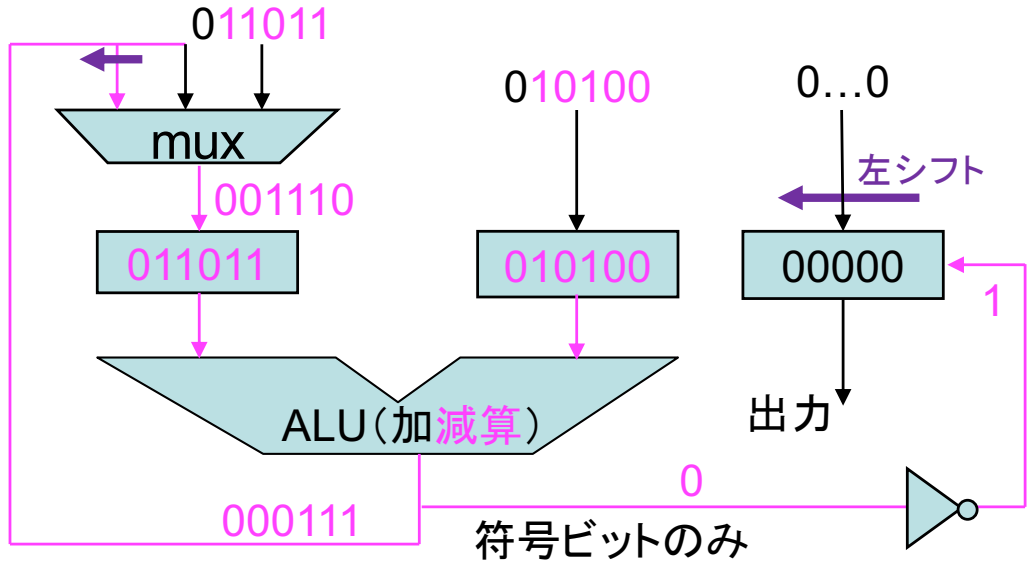
- 繰り返し型除算器のハードウェア構成です。
  - Qレジスタ(シフトレジスタを使用)に商を格納します。最初に0に初期化し、商が1ビット求まるごとに、それを右端から挿入して左へ1ビットシフトします。
  - Yレジスタには除数Yをセットします。
  - Rレジスタには余りを格納します。最初に被除数Xで初期化します。



- 動作例
  - サイクル1
    - R、Yレジスタに入力し、Qレジスタを初期化します。

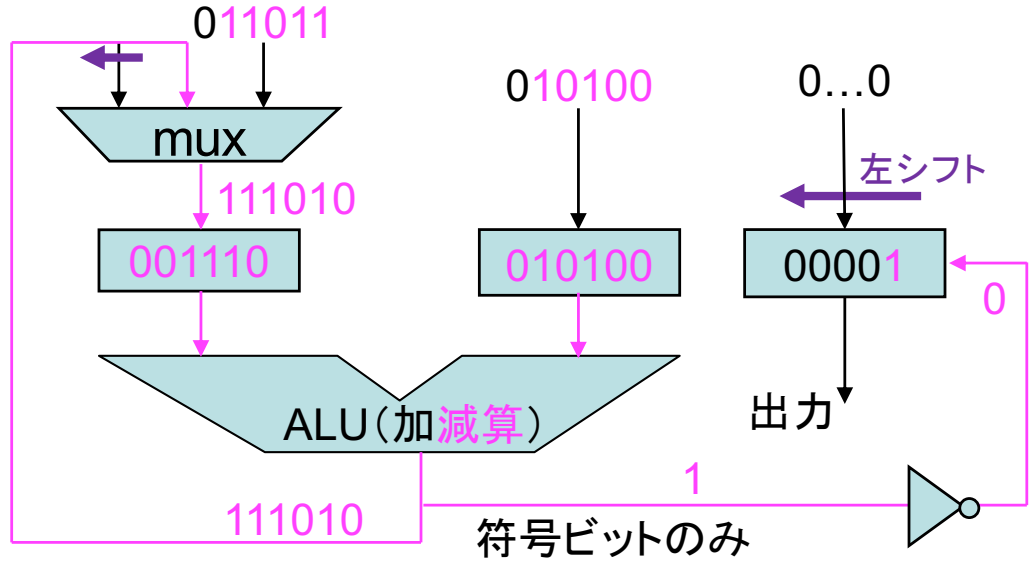


  - サイクル2
    - Rレジスタの内容からYレジスタの内容を引き（結果は正）、Qレジスタを更新します。



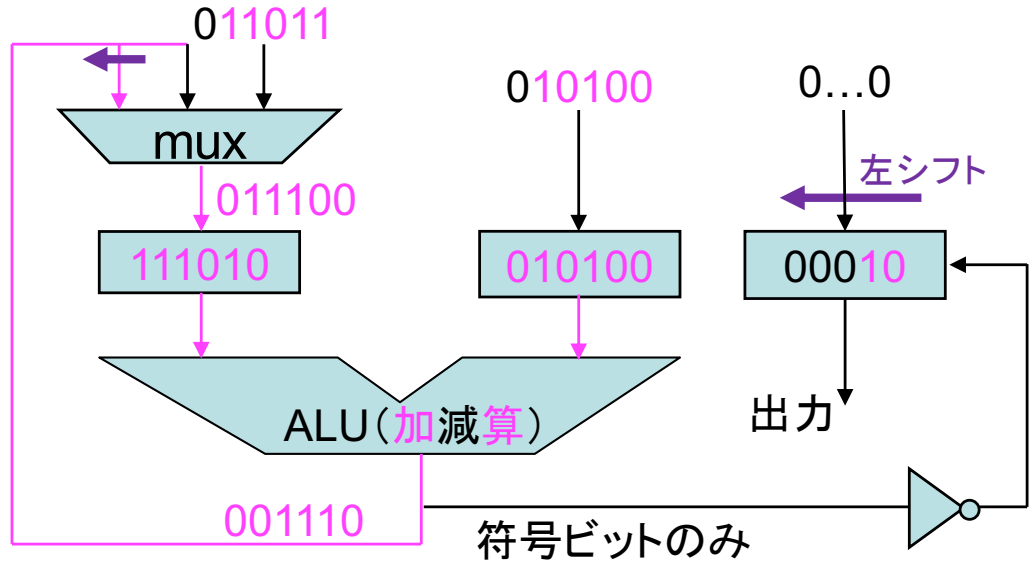
– サイクル3

- Rレジスタの内容からYレジスタの内容を引き（結果は負）、Qレジスタを更新します。



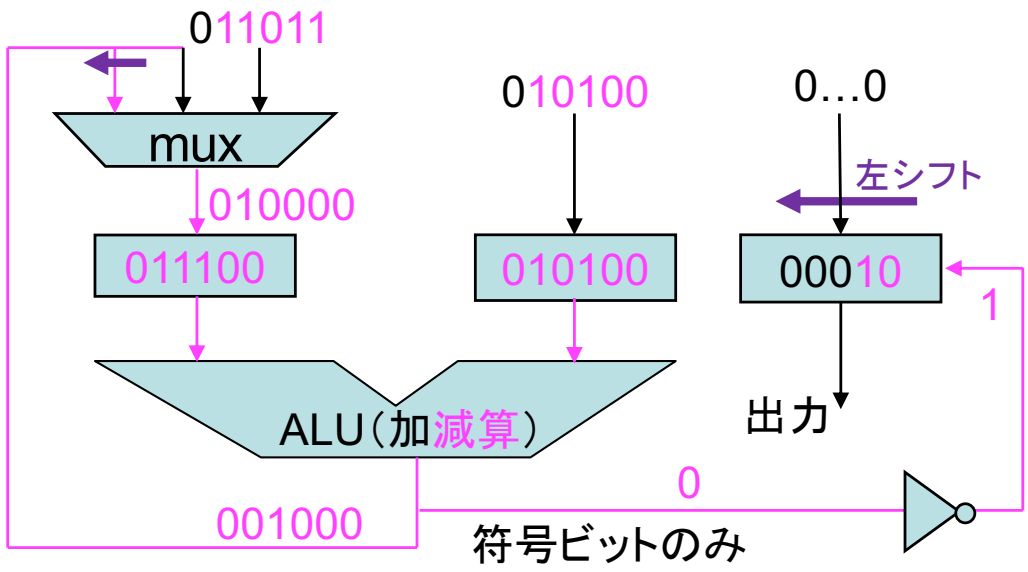
– サイクル4

- Rレジスタの内容にYレジスタの内容を足して、元に戻してシフトます。



– サイクル5

- Rレジスタの内容からYレジスタの内容を引き（結果は正）、Qレジスタを更新します。



– サイクル6以降

（省略）

- 商を1ビット求めるのに1サイクル（減算）かかるのは仕方ないとして、2サイクル（減算と加算）かかる場合があります。
  - ハードウェア構成を少し工夫すれば、引き戻しのために加算しなくてもよくなるのですが、一応、昔の人は、1ビット求めるのに1または2サイクルかかるとしていました。
  - しかし、そのおかげで、1ビット求めるのに必ず1サイクルしかかからないアルゴリズムを考え出しました。それが、次に解説する「引き放し法」です。
    - 単にハードウェア構成を最適化するのとは異なり、アルゴリズム的に1サイクルで1ビット求めることを可能にしました。

引き放し法

- 引き戻し法では、余りが負になった場合に除数を加えて元に戻していましたが、引き放し法では、除数を加えて元に戻すことなく、余りが負になったままで商の次のビットを求めます。
  - 1サイクルで商の1ビットを求めます。そして、1サイクルの間に減算と加算のどちらか一方のみを行います。
- 引き放し法の原理を、算数レベルで理解することも不可能ではありません。しかし、それでは応用が利かないので、本講義では、数学的な基礎を踏まえて、引き放し法の原理を理解することを試みます。

引き放し法の概要

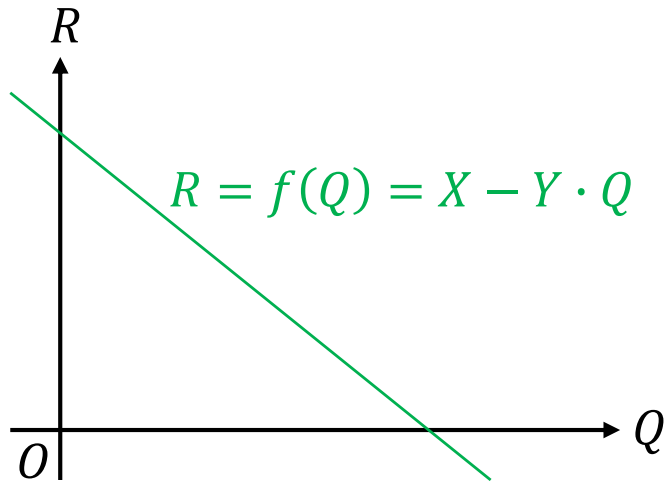
- 以下のように変数を定義します。
  - X：被除数
  - Y：除数
  - Q：商
  - R：余り（誤差）
- このとき、次の関係が成り立ちます。

$$X = Y \cdot Q + R \Leftrightarrow R = X - Y \cdot Q$$

- よって、Qは以下の方程式の解となります。

$$0 = X - Y \cdot Q$$

- したがって、この方程式をどうやって解くかが問題になります。
- そのために、まず、RとQの関係をグラフで考えます。

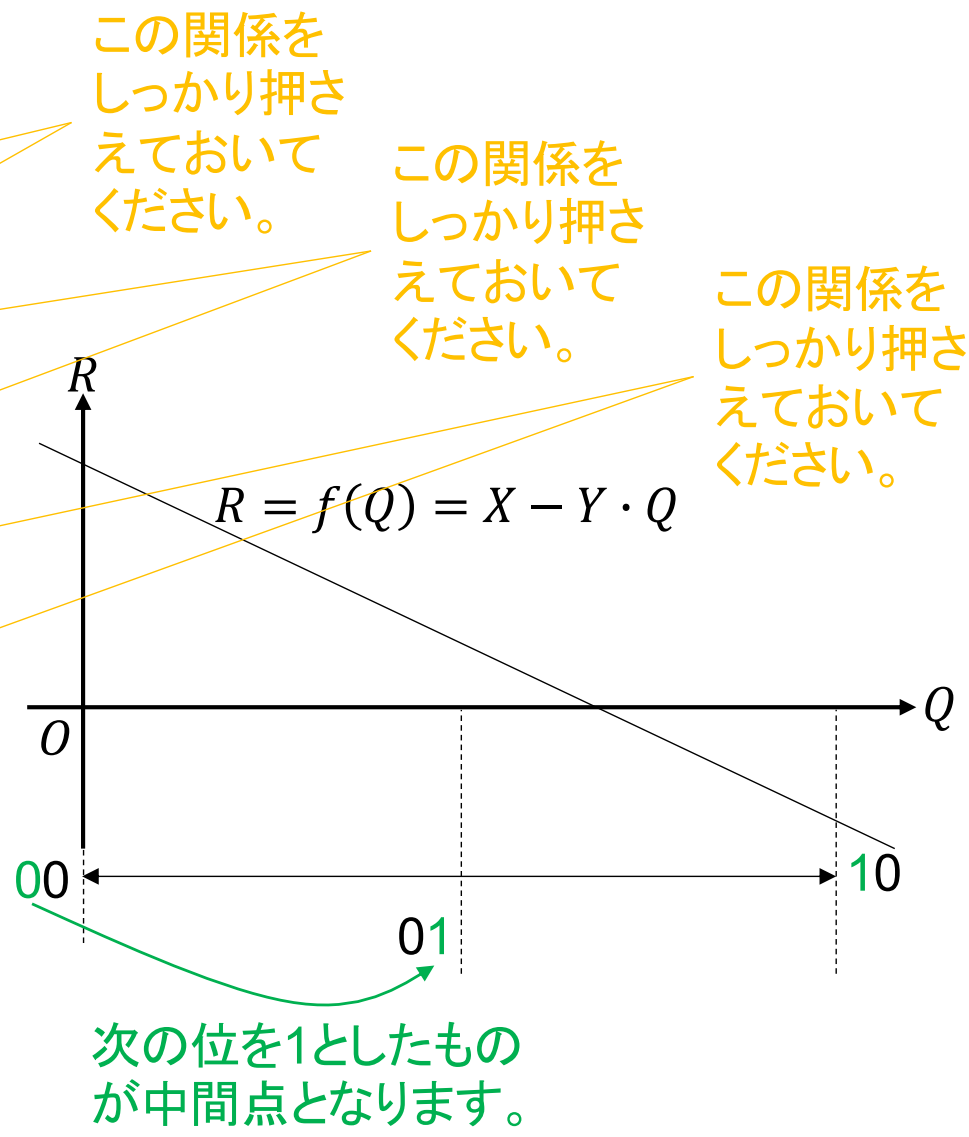


- そして、$R = f(Q)$の直線と軸との交点の座標を二分法によって求めていきます。

二分法とは？

- 解を含む区間を、その中点で2つに分割することによって、解の存在する範囲を1/2（半分）に狭めていく方法です。
- 2進数との親和性が高いという特徴があり、これを利用して方程式の解を求めます。
  - 2進数で1の半分は0.1です。
  - 例えば、解が1.101と1.110の間にあるとします（この領域の幅は1.110−1.101=0.001で、小数点第3位の解像度であることに注意してください）。この中点は1.1011です。つまり、この区間の左端である1.101に対して、その次の位を1にしたものが、中点になります。

- まず、解が0と10(=2)の間にあることがわかっているとします。
- 0と10の中点は1です。
  - 解が0と1の間にあるならば、解は「0.～」となります。
  - 解が1と10の間にあるならば、解は「1.～」となります。
- 右図の例では、解は「1.～」となります。整数部が求まりました。

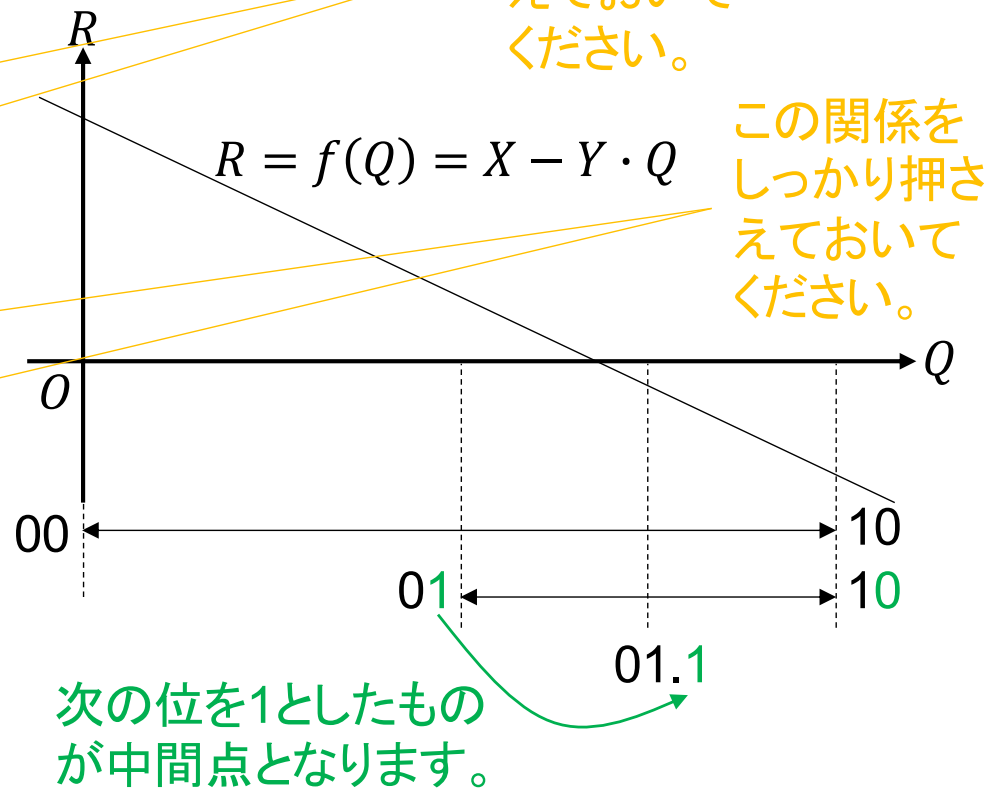


- 次の桁を求めます。
- 解は1と10(=2)の間にあります。
- 1(=1.0)と10(=10.0)の中点は1.1です。
  - 解が1.0と1.1の間にあるならば、解は「1.0～」となります。
  - 解が1.1と10.0の間にあるならば、解は「1.1～」となります。
- 右図の例では、解は「1.0～」となります。

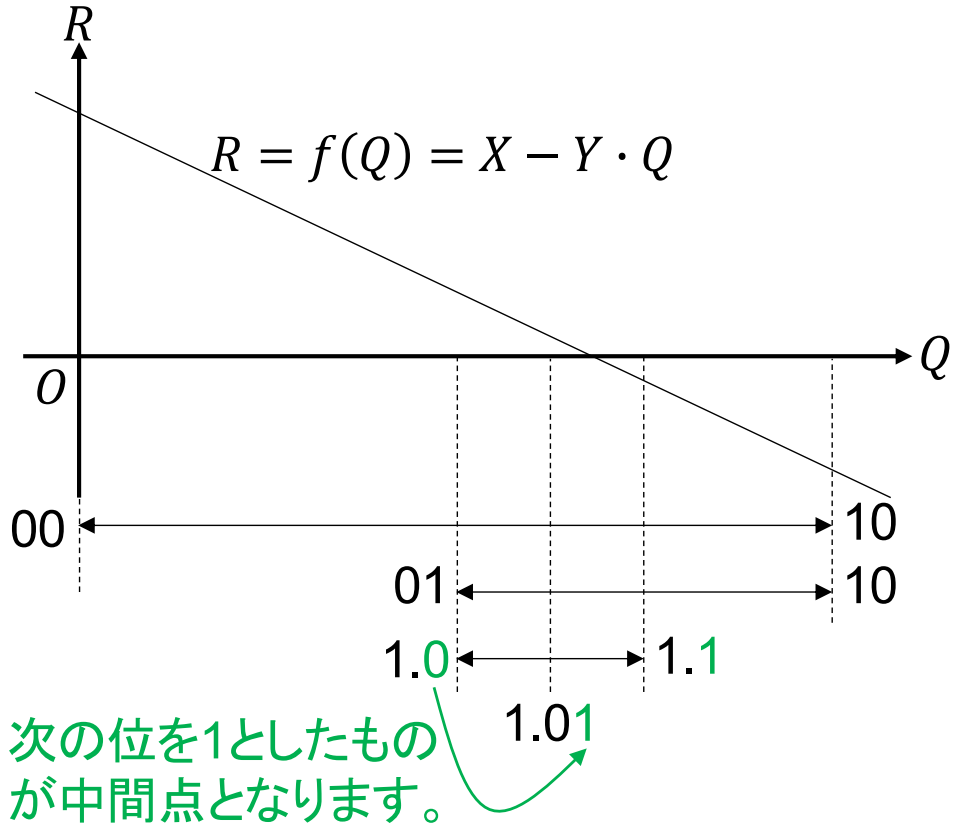
この関係をしっかり押さえておいてください。

この関係をしっかり押さえておいてください。

この関係をしっかり押さえておいてください。



- 次の桁を求めます。
- 解は1.0と1.1の間にあります。
- 1.0と1.1の中点は1.01です。
  - 解が1.00と1.01の間にあるならば、解は「1.00～」となります。
  - 解が1.01と1.10の間にあるならば、解は「1.01～」となります。
- 右図の例では、解は「1.01～」となります。



- ここまでを整理しておきます。
  - 小数第k位まで求めた解（部分解）をQ_kとし、その各桁を$q_i(0 \leq i \leq k)$として以下のように表現することにします。

2^{-k}の位

$$Q_k = [q_0.q_1 q_2 \cdots q_{k-1} q_k]$$

  - 小数第k位まで解が求まっているとき、本当の解はQ_kと$(Q_k + 2^{-k})$の間にあります。
  - Q_k～$(Q_k + 2^{-k})$の区間の中点M_{k+1}は

$$M_{k+1} = \frac{Q_k + (Q_k + 2^{-k})}{2} = Q_k + 2^{-(k+1)}$$

つまり、

$2^{-(k+1)}$の位

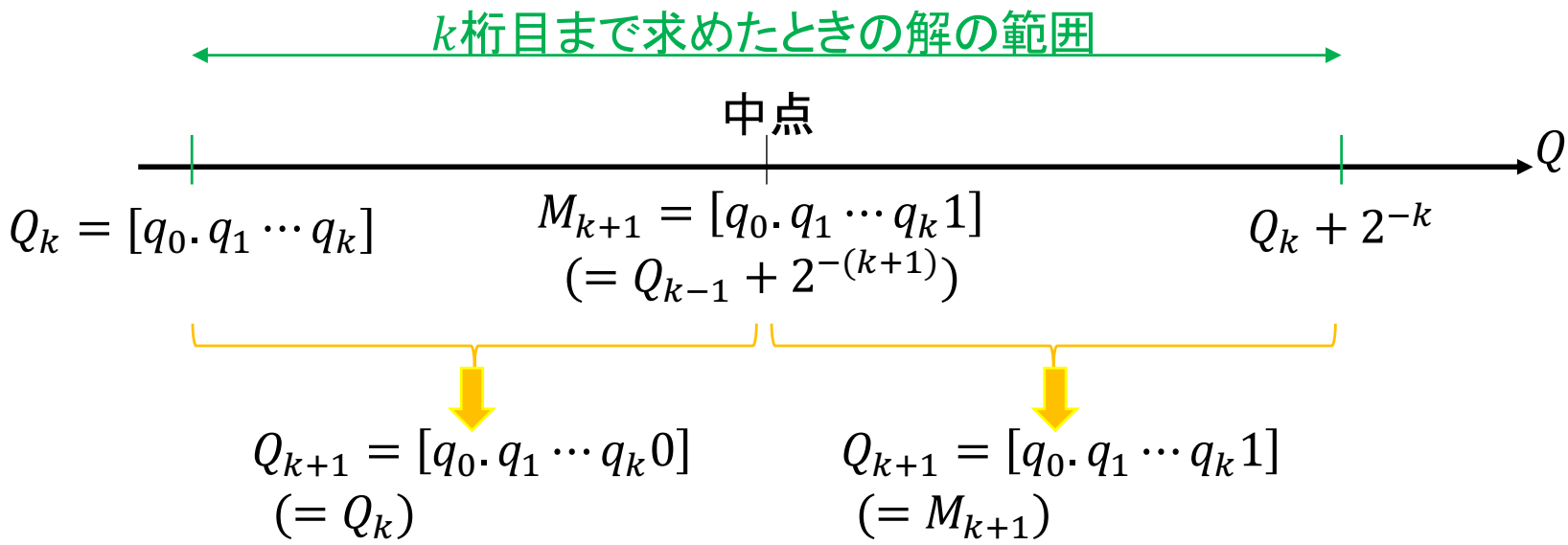
$$M_{k+1} = [q_0.q_1 q_2 \cdots q_{k-1} q_k 1]$$

– そして、次にQ_{k+1}（つまりq_{k+1}）を求めます。
– 本当の解が中点M_{k+1}の左の区間にあるなら、

$$Q_{k+1} = [q_0.\, q_1 q_2 \cdots q_{k-1} q_k 0]\ (= Q_k)$$

– 本当の解が中点M_{k+1}の右の区間にあるなら、

$$Q_{k+1} = [q_0.\, q_1 q_2 \cdots q_{k-1} q_k 1]\ (= M_{k+1})$$



- 以上のように、商の次の（下の）位の値を求めることは、半分に分割した区間のどちらに解が含まれるのかを判定することと対応します。
- では、解が左右のどちらの区間に含まれるかを判定するのは、どうすればよいでしょうか？

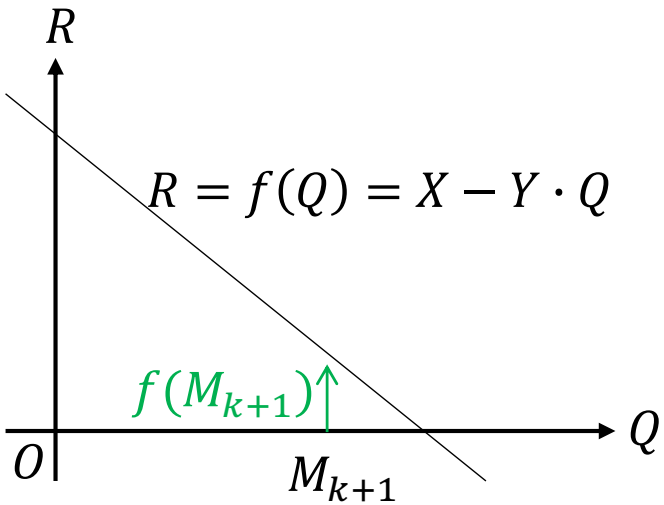
- その判定を簡単に行うために、「中間値の定理」を用います。

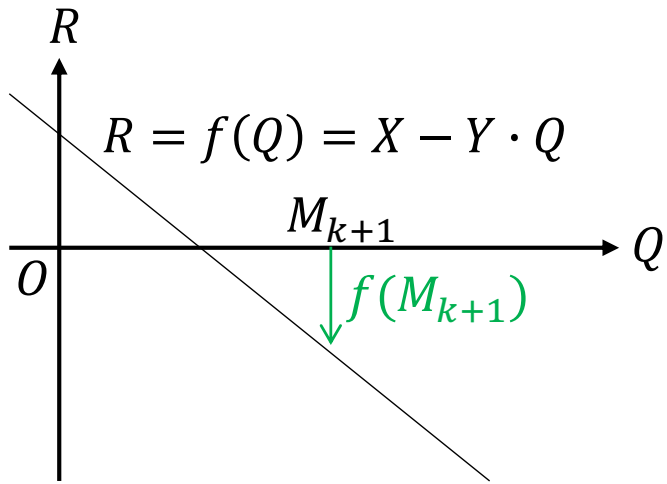
中間値の定理の利用

- 「中間値の定理」は高校の数学で習っていますね。
  - 難しい定理ではありませんが、忘れているなら復習して思い出してください。

- さて、$R = f(Q) = X - Y \cdot Q$のグラフは傾きが負$(-Y)$の直線です。
- つまり、$f(Q)$は単調減少関数です。

- したがって、中点の座標値M_{k+1}に対して、$f(M_{k+1})$ の符号（正負）で、以下のように判定できます。
  - $f(M_{k+1}) \geq 0$なら解は右にある。
  - $f(M_{k+1}) < 0$なら解は左にある。





引き放し法の計算原理

- 以上で、引き放し法でどのように割り算の商を求めようとしているのか、おおよその戦略が理解できたはずです。

- 残る課題は、「$f(M_{k+1})$の値をどのようにして求めるか」です。
  - $f(M_{k+1}) = X - Y \cdot M_{k+1}$をそのまま計算したのでは話になりません（乗算が必要になってしまいます）。
  - 1サイクルで商を1ビット求める、また、1ビット求めるのに加減算を1回しか行わない、というのが当初の目的でした。

- 次ページ以降で、引き放し法による計算方法の原理を、(1)～(3)の順に解き明かしていきます。

  - その前に、余りRに関して、以下のように定義しておきます。

$$R_k = f(M_k) = X - Y \cdot M_k$$
$$r_k = 2^k \cdot R_k$$

    - 実際の商の計算では、R_kそのものではなく、R_kを2^k倍したr_kを計算します。
    - R_kとr_kの符号(正負)は同じであることに注意してください。

(1) 解の範囲確認

- 正規化数を前提とすると、

$$X = [1.\cdots], Y = [1.\cdots]$$

つまり

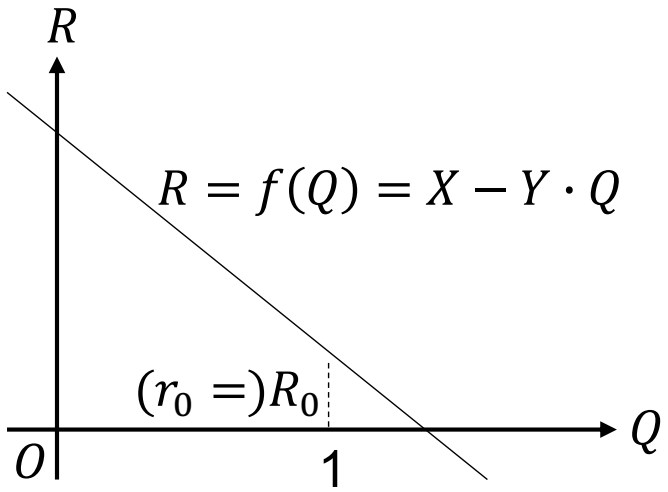
$$1 \leq X < 2$$

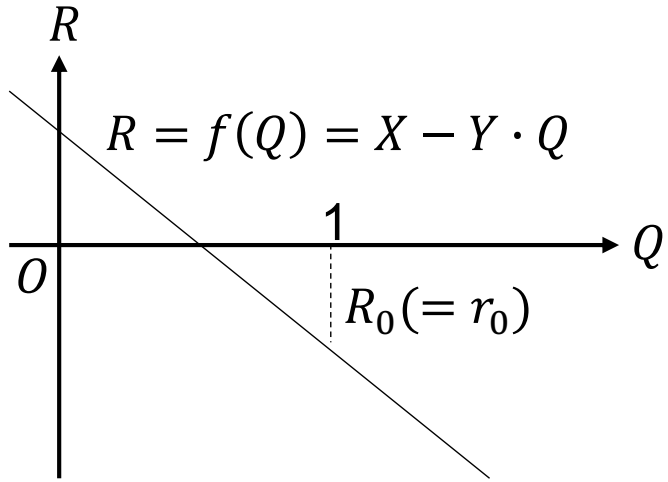
$$1 \leq Y < 2 \Leftrightarrow \frac{1}{2} < \frac{1}{Y} \leq 1$$

- 以上より、 $\frac{1}{2} < \frac{X}{Y} < 2 \quad (\Longrightarrow 0 < \frac{X}{Y} < 2)$
  - よって、商は $\frac{X}{Y} = [0.1\cdots]$ または $\frac{X}{Y} = [1.\cdots]$
    つまり、整数部は1桁です。

(2) 整数部の計算

- 商の整数部が0か1かを確かめます。
  - $M_0 = 1$として
    $(r_0 =)R_0 = f(M_0) = X - Y \cdot M_0 = X - Y$
  - $R_0 \geq 0$ なら q_0=1、$R_0 < 0$ なら q_0=0です。



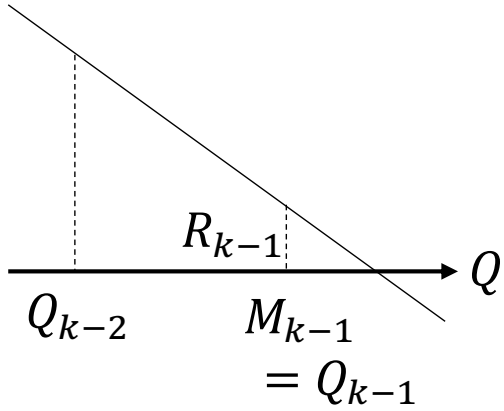


(3) 小数部の計算

- 小数第$k-1$位$(k \geq 1)$までの商Q_{k-1}が求まったとき、次にq_kを求めます。
  - $Q_{k-1} = [q_0.q_1 q_2 \cdots q_{k-1}]$
  - $M_k = [q_0.q_1 q_2 \cdots q_{k-1}1] = Q_{k-1} + 2^{-k}$
  - よって、
    $R_k = X - Y \cdot M_k = X - Y \cdot (Q_{k-1} + 2^{-k})$
  - ここで、次の2つのケースに分けて考えます。
    (3a) $R_{k-1} \geq 0$ だったので $q_{k-1} = 1$ とした場合
    (3b) $R_{k-1} < 0$ だったので $q_{k-1} = 0$ とした場合

(3a) $R_{k-1} \geq 0 (q_{k-1} = 1)$の場合

- $M_{k-1} = [q_0 . q_1 q_2 \cdots q_{k-2} 1]$
- $Q_{k-1} = [q_0 . q_1 q_2 \cdots q_{k-2} 1]$
- よって、$M_{k-1} = Q_{k-1}$



- これより

$$R_{k-1} = X - Y \cdot M_{k-1} = X - Y \cdot Q_{k-1}$$

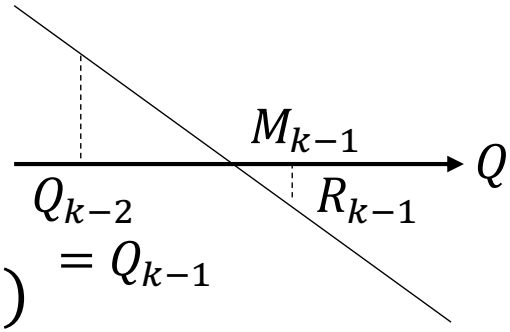
- ここで、先のR_kからこのR_{k-1}を引きます。

$$R_k - R_{k-1} = \cdots$$（次ページへ続く）

- $R_k - R_{k-1}$
  $= \left\{X - Y \cdot \left(Q_{k-1} + 2^{-k}\right)\right\} - \{X - Y \cdot Q_{k-1}\}$
  $= -Y \cdot 2^{-k}$
- よって、$R_k = R_{k-1} - Y \cdot 2^{-k}$
  - 両辺に2^kを掛けて
    $2^k R_k = 2 \cdot 2^{k-1} R_{k-1} - Y \Leftrightarrow r_k = 2r_{k-1} - Y$
  - つまり、r_{k-1}を左に1ビットシフトしたものからYを引いて、r_kとします。
- その結果、以下のようにq_kを求めます。
  - $r_k \geq 0$ $(R_k \geq 0)$ なら $q_k = 1$
  - $r_k < 0$ $(R_k < 0)$ なら $q_k = 0$

(3b) $R_{k-1} < 0 (q_{k-1} = 0)$の場合

- $M_{k-1} = [q_0.\, q_1 q_2 \cdots q_{k-2} 1]$
- $Q_{k-1} = [q_0.\, q_1 q_2 \cdots q_{k-2} 0]$
- よって、$M_{k-1} = Q_{k-1} + 2^{-(k-1)}$



- これより

$$R_{k-1} = X - Y \cdot M_{k-1}$$
$$= X - Y \cdot (Q_{k-1} + 2^{-(k-1)})$$

- ここで、先のR_kからこのR_{k-1}を引きます。

$$R_k - R_{k-1} = \cdots$$（次ページへ続く）

- $R_k - R_{k-1}$
  $= \{X - Y \cdot (Q_{k-1} + 2^{-k})\}$
  $\quad - \{X - Y \cdot (Q_{k-1} + 2^{-(k-1)})\}$
  $= -Y \cdot 2^{-k} + Y \cdot 2 \cdot 2^{-k} = Y \cdot 2^{-k}$
- よって、$R_k = R_{k-1} + Y \cdot 2^{-k}$
  - 両辺に2^kを掛けて
    $2^k R_k = 2 \cdot 2^{k-1} R_{k-1} + Y \quad \Leftrightarrow r_k = 2r_{k-1} + Y$
  - つまり、r_{k-1}を左に1ビットシフトしたものにYを足して、r_kとします。
- その結果、以下のようにq_kを求めます。
  - $r_k \geq 0$ $(R_k \geq 0)$ なら $q_k = 1$
  - $r_k < 0$ $(R_k < 0)$ なら $q_k = 0$

引き放し法による除算アルゴリズム

- 以上をまとめると、以下の手続きで商が求まります。
  ① $k = 0$ とし、$r_0(= 2^0 \cdot R_0) = X - Y$ を求めて②へ進む。
  ② $r_k \geq 0$ なら $q_k = 1$、
  $r_k < 0$ なら $q_k = 0$、とし、③へ進む。
  ③ 割り切れるか、または必要桁数の商が得られれば終了。そうでなければ、kを1増やして④へ進む。
  ④ 以下の式を用いて r_k を求め、②へ戻る。
    - $r_{k-1} \geq 0$ $(q_{k-1} = 1)$ なら　$r_k = 2r_{k-1} - Y$
    - $r_{k-1} < 0$ $(q_{k-1} = 0)$ なら　$r_k = 2r_{k-1} + Y$

- 計算例：　1.1011÷1.0100