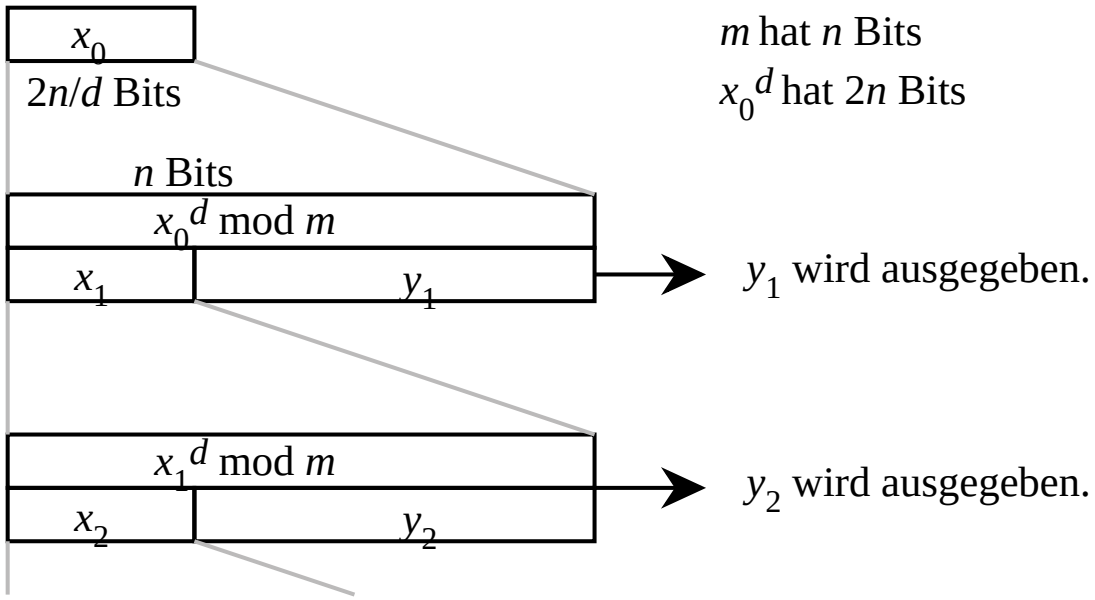
4.5 Der Micali-Schnorr-Generator

Der von Micali und Schnorr vorgeschlagene Zufallsgenerator ist ein Abkömmling des RSA-Generators. Sei dazu $d \geq 3$ ungerade. Als Parametermenge dient die Menge aller Produkte m von zwei Primzahlen p und q, die sich in ihrer Bitanzahl höchstens um 1 unterscheiden und für die d zu $\varphi(m) = (p-1)(q-1)$ teilerfremd ist. Wenn m eine n-Bit-Zahl ist, sei $r(n) \approx \frac{2n}{d}$; die d-te Potenz einer $r(n)$-Bit-Zahl ist dann (ungefähr) eine $2n$-Bit-Zahl.

Im i-ten Schritt wird $z_i = x_{i-1}^d \bmod m$ gebildet; davon werden die ersten $r(n)$ Bits, also $\lfloor z_i / 2^{n-r(n)} \rfloor$, als x_i genommen, die übrigen Bits, also $y_i = z_i \bmod 2^{n-r(n)}$ werden ausgegeben. Bemerkenswert ist, dass die Bits auf zwei *disjunkte* Teile verteilt werden: den Wert x_i für den nächsten Schritt und die Ausgabe y_i. Die folgende Abbildung macht das deutlich.



Ist G perfekt? Hier wird folgendes angenommen: Kein effizienter Test kann die Gleichverteilung auf $[1 \ldots m]$ von der Verteilung von $x^d \bmod m$ für gleichverteiltes $x \in [1 \ldots 2^{r(n)}]$ unterscheiden. Ist diese Annahme richtig, so ist der Micali-Schnorr-Generator perfekt.

Es scheint auf den ersten Blick fast, als ob hier nur die Perfektheit unter der Annahme der Perfektheit bewiesen wird; allerdings gibt es heuristische Überlegungen, die die Annahme in enge Beziehung zur Sicherheit des RSA-Verschlüsselungsverfahrens und zur Primzerlegung bringen.

Wie schnell purzeln nun die Zufallszahlen aus der Maschine? Als elementare Operationen gezählt werden sollen wieder die Multiplikation zweier 32-Bit-Zahlen und die Division einer 64-Bit-Zahl durch eine 32-Bit-Zahl mit

32-Bit-Quotient. Multipliziert und dividiert wird nach der klassischen Methode; das Produkt von r (32-Bit-)Wörtern mit s Wörtern kostet also rs elementare Operationen, bei der Division ist der Aufwand das Produkt der Wortzahlen von Divisor und Quotient. Die Multiplikation mit der schnellen Fourier-Transformation bringt erst bei größeren Stellenzahlen Vorteile.

Die Erfinder machen nun einen konkreten Vorschlag: $d = 7$, $n = 512$. Ausgegeben werden jeweils 384 Bits, zurückbehalten werden 128 Bits. Das binäre Potenzieren einer 128-Bit-Zahl x mit 7 kostet eine Reihe elementarer Operationen:

- x hat 128 Bits, also 4 Wörter.
- x^2 hat 256 Bits, also 8 Wörter, und kostet $4 * 4 = 16$ elementare Operationen.
- x^3 hat 384 Bits, also 12 Wörter, und kostet $4 * 8 = 32$ elementare Operationen.
- x^4 hat 512 Bits, also 16 Wörter, und kostet $8 * 8 = 64$ elementare Operationen.
- x^7 hat 896 Bits, also 28 Wörter, und kostet $12 * 16 = 192$ elementare Operationen.
- $x^7 \bmod m$ hat ≤ 512 Bits und kostet ebenfalls $12*16 = 192$ elementare Operationen.

Insgesamt braucht man also 496 elementare Operationen; es war nur eine Reduktion $\bmod m$ nötig. Die Belohnung besteht aus 384 Bits. Man erhält also 32 Bits mit etwa 40 elementaren Operationen. Damit hat man gegenüber den linearen Kongruenzgeneratoren, wenn man dort alle 32 Bits verwendet, nur einen Faktor 10, den heutige Computer gut verkraften.

Eine fast beliebige Geschwindigkeitssteigerung ergibt sich durch Parallelisierung: Der Micali-Schnorr-Generator ist vollständig parallelisierbar; das bedeutet, dass eine Verteilung der Arbeit auf k Prozessoren einen Gewinn um den echten Faktor k bedeutet: Die Prozessoren können unabhängig voneinander arbeiten ohne Notwendigkeit zur Kommunikation.