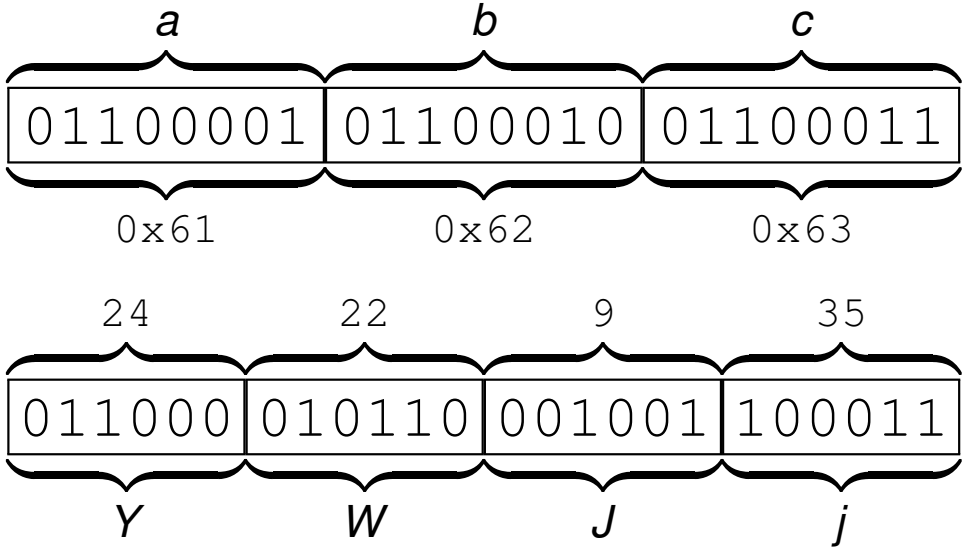
Base64-Codierung

Zweck: Vermeidung nicht druckbarer Zeichen unter Beibehaltung des Informationsgehalts

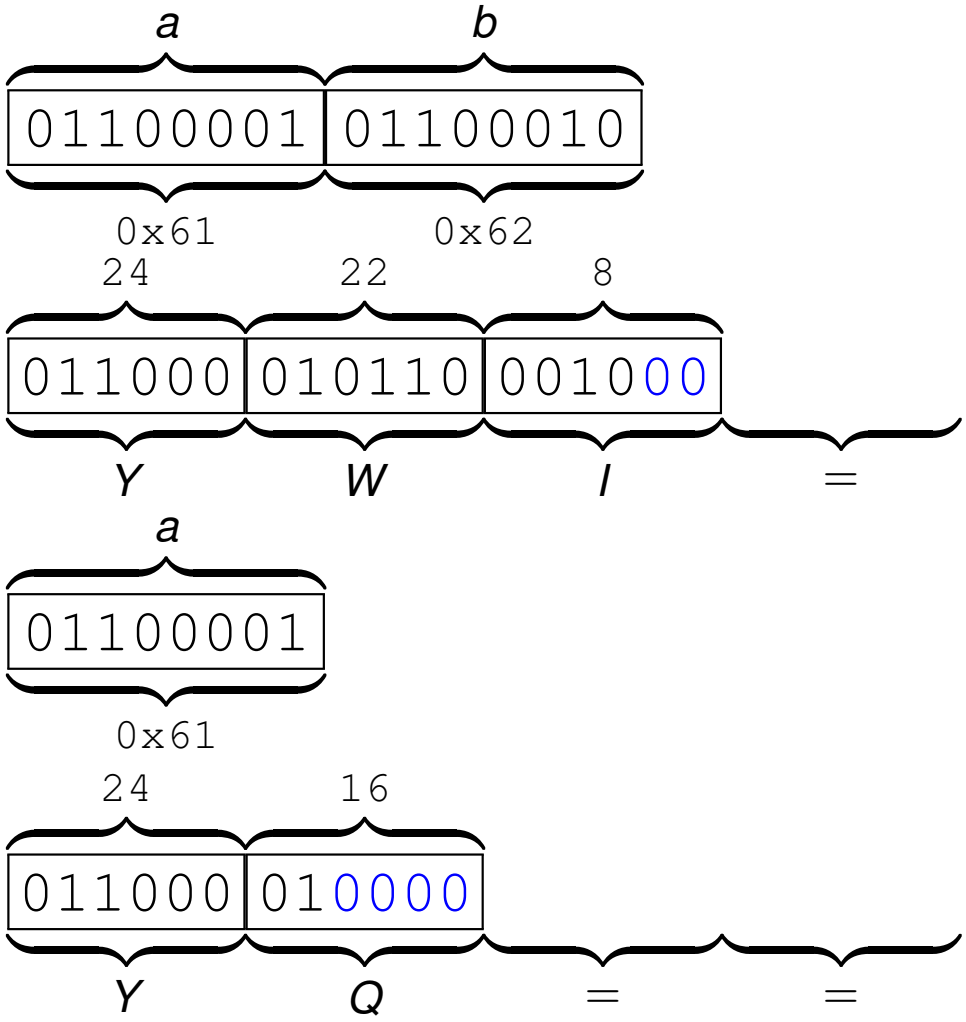
Bem.: Vergrößerung der Datenmenge um ein Drittel

- 3 Byte $\hat{=}$ 24 Bit $\hat{=}$ 4 6-Biteinheiten
- Jede 6-Biteinheit (insgesamt 64 Stück) bekommt ein druckbares Zeichen (`A-Z a-z 0-9 + /`) zugeordnet.



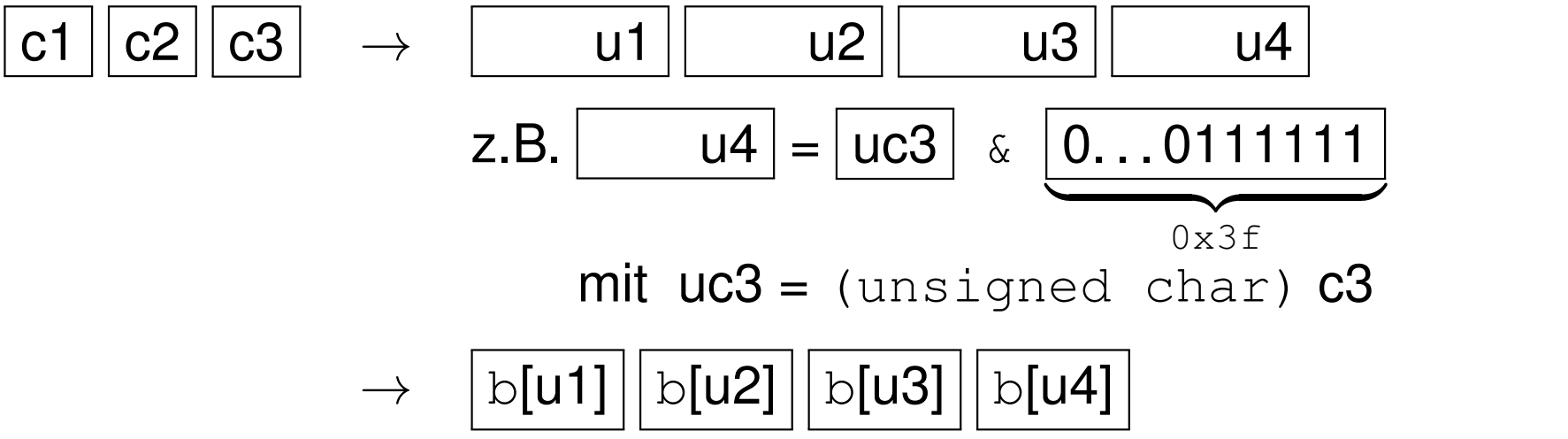
Base64-Codierung - Fortsetzung

- Unvollst. 6-Biteinheiten werden mit Nullbits aufgefüllt.
- Fehlende 6-Biteinheiten in einer Vierergruppe werden durch = dargestellt.



Hinweise zur Programmierung: Binär → Base64

Anfangsbuchstaben	*Datentyp*
b c	`char`
uc	`unsigned char`
u	`unsigned int`



- Jeweils 3 Zeichen von `ein` auf C-Vektor `cv` einlesen:
  `char cv[3]; ein.read(cv,3); nread=ein.gcount()`
  `nread` enthält Anzahl der tatsächlich gelesenen Zeichen seit letztem `ein.read`. Dementsprechend Zuweisungen `c1=cv[0]; c2=cv[1]; c3=cv[2];` vornehmen

Weitere Hinweise zur Programmierung

Vorübung

- Erstellen Sie ein Programm, das die Zeichen in der Binärdatei `binaer.dat` zählt.

Testmöglichkeiten, Fallstricke

- An Linuxrechnern können Sie evtl. das Kommando `base64` zum Testen verwenden. (Ggf. Umleiten in eine Datei.)
- Nützlich kann an Linuxrechnern das Kommando `od -c` zur Anzeige nicht druckbarer Zeichen bzw. Steuerzeichen in einer Datei sein.
- In Ausdrücken findet Integererweiterung statt, insb. werden auch `char` bzw. `unsigned char` in `int` konvertiert.