sysjm

Beschreibung der Programmfamilie

CODING

Programme zur direkten Chiffrierung

Benutzer-Handbuch

Dipl.-Math. Jürgen Müller

Version 1.1 / Stand: 01.12.2013

1 Beschreibung der notwendigen Parameter des Programms CODING 4
1.1 Anstarten 4
1.2 Sprache 5
1.3 Identifikation 5
1.4 Einschränkungen 6
1.5 Allgemeines 6
1.6 Kontrollausgabedatei 6
1.7 Ausgabe der Kommandozeilen-Steuerungs-Parameter 7
1.8 Parameterdatei-Bearbeitung 7
1.8.1 Parameterdatei 8
1.8.2 Parameterdatei-Verwendung 9
1.8.3 Chiffrierung nach Parameterdatei-Behandlung 9
1.8.4 Zeitgrenze 9
1.8.5 Aktion bei Zeitgrenzen-Überscheitung 9
1.9 Angaben zu den zu behandelnden Dateien 10
1.10 Ausnahmedatei 11
1.11 Bearbeitungsform 12
1.12 Statistik 12
1.13 Schlüssel-Informationen 13
1.13.1 Abfrage zur Parameterdatei-Sicherung 14
1.13.2 Zeichensatz 15
1.13.3 Eingabe-Modus 15
1.13.4 Tastatur-Eingabe 15
1.13.5 Eingabe via Schlüssel-Datei 16
1.13.6 Zufallsschlüssel-Eingabe 17

2 Beschreibung der Überladeinformationen von CODING 18
2.1 Überladeinformation 18
2.2 Stellen-Intervall 19
2.3 Überladewerte-Intervall 19
2.4 Schlüsselwerte-Intervall 20
2.5 Eingabe-Modus 21
2.6 Tastatur-Eingabe 21
2.7 Eingabe via Überlade-Dateiangaben 22
2.8 Überladedateien im Direktmodus 22

3 Eigentliche Programm-Bearbeitung 24
3.1 Programm-Initialisierung 24
3.2 Programm-Chiffrierung 26

4 Erweiterungen des Standard-Programms 28
4.1 Stromchiffrierung 28
4.2 Komponentenwechsel-Parameterdatei 29
4.3 Stromchiffrierungs-Behandlungswert 30
4.4 Kompromittierungs-Schlüssel 30

4.5 Störungen des Datenaustauschs....31
4.6 Weitere potentielle Anwendungsmöglichkeiten....31

5 System- und Algorithmus-Parameter des Programms....32
5.1 System-Parameter....32
5.1.1 Gruppe....32
5.1.2 Modulart....32
5.1.3 DateiSystem....32
5.1.4 ZeichenSatz....32
5.1.5 BSigRtL....32
5.1.6 Verpflichtung....32
5.2 Algorithmus-Parameter....33
5.2.1 varBlk (CODING1/2)....33
5.2.2 BlockElem....33
5.2.3 BlkKey....33
5.2.4 BlkKey2 (CODING3)....33
5.2.5 Safety....33
5.2.6 stLoop....34
5.2.7 stLoopC....34
5.2.8 addLoop....34
5.2.9 opLoop....34
5.2.10 PDLoop....34
5.2.11 EndShift....35
5.2.12 ClearCore....35
5.2.13 MeanBlksX....35
5.2.14 MeanBlksD....35
5.2.15 MeanBlksO....35

6 Kontakt-Daten....35

1 Beschreibung der notwendigen Parameter des Programms CODING

1.1 Anstarten

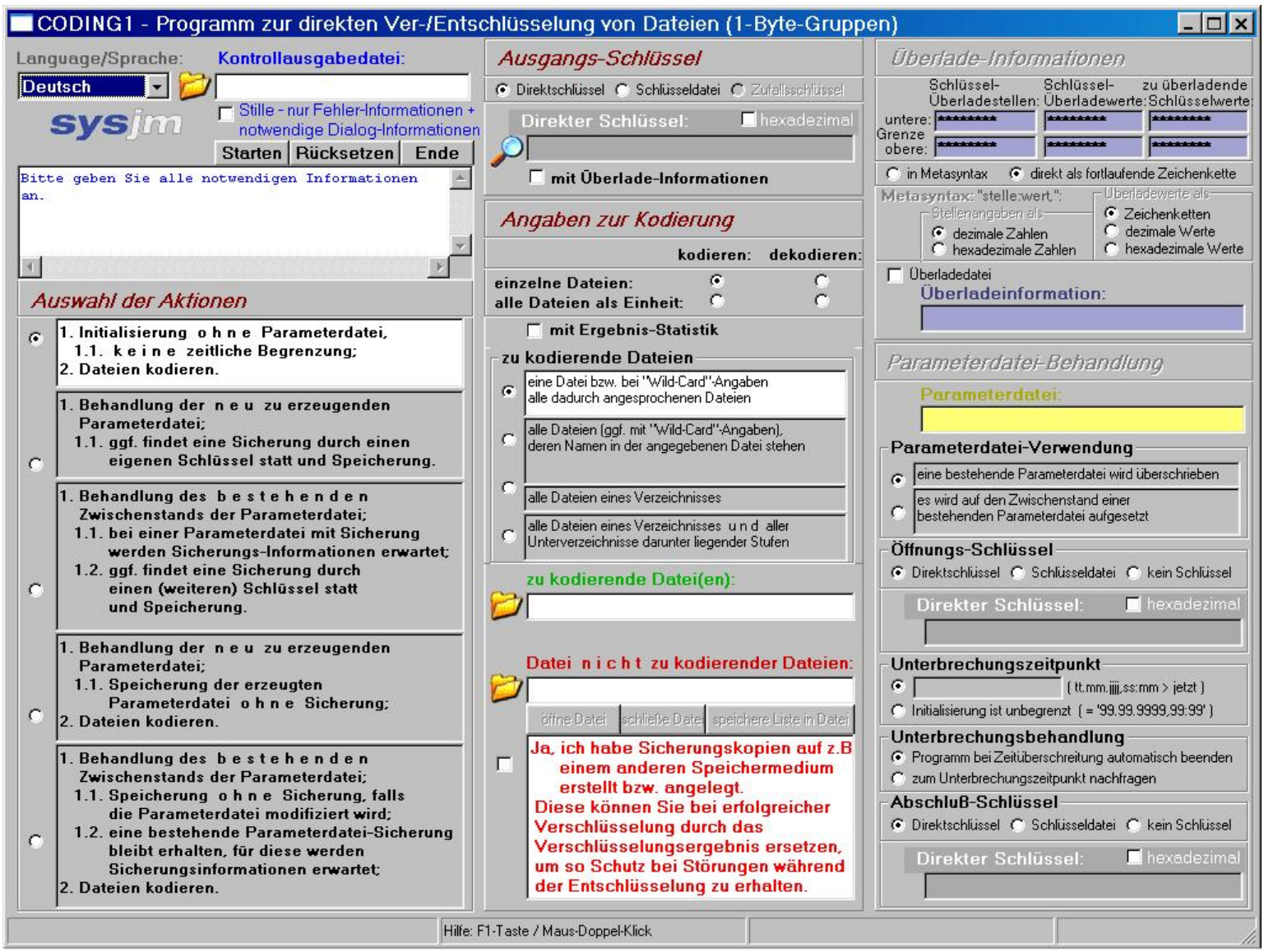
Bat: Sie können eines der **DOS-Programme** (CODINGn.EXE, n=1,2,3) direkt aus MS-Windows heraus aufrufen (z.B. durch Doppel-Klick), worauf das Programm innerhalb einer DOS-Box gestartet wird. Im weiteren werden Angaben zu DOS-Programmen „**Bat:**“ vorangestellt.

Dia: Um eines der entsprechenden **MS-Windows-Programme** anstarten zu können (vorangestellte Kennzeichnung „**Dia:**“), ist die entsprechende Datei CODING.EXE in das Verzeichnis zu kopieren, in dem die entsprechenden Windows-Programme „entpackt“ werden sollen. Danach ist das Entpacken durch Aufruf von CODING.EXE zu veranlassen.

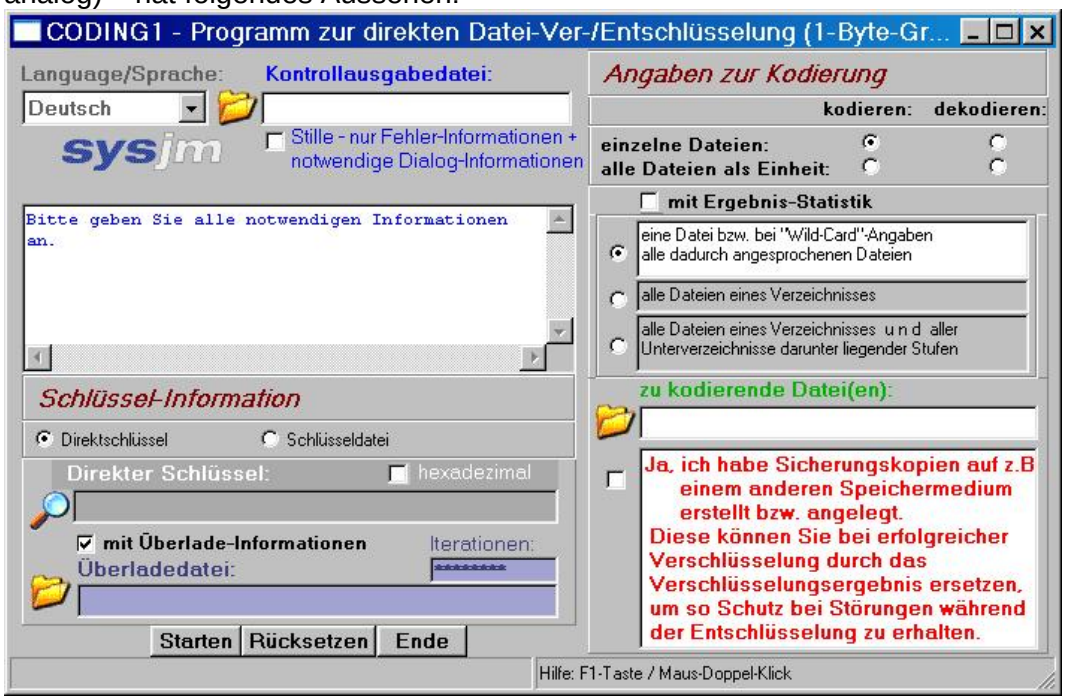
Beachten Sie, dass dabei im Ausgangsverzeichnis Unterverzeichnisse HELPDBx (z.Zt. x=’D’ und x=’E’) mit Hilfetexten der einzelnen Sprachen erstellt werden.

Danach können Sie eines der entsprechenden MS-Windows-Programme zum einen in der „einfachen“ Version (CODINGnS.EXE, n=1,2,3) bzw. zum anderen in der „erweiterten“ Version (CODINGnE.EXE, n=1,2,3) direkt anstarten.

Nach dem Anstarten des Programms wird das Haupteingabeformular – hier zunächst in der erweiterten Version von CODING1E (für CODING2E/3E analog) – angezeigt:



Das Haupteingabeformular für die „einfache“ Version – hier von CODING1S (für CODING2S/3S analog) – hat folgendes Aussehen:



1.2 Sprache

Bat: Zunächst werden Sie aufgefordert anzugeben, in welcher Sprache Sie den nachfolgenden Dialog wünschen (hier beispielhaft für Englisch und Deutsch):

```
Language:
  English    --> 0
  Deutsch    --> 1
```

Geben Sie die Ziffer ein, dessen zugehörige Sprache von Ihnen gewünscht wird.

Dia: Bei Eingabe der Parameter können Sie jeder Zeit die Sprache wechseln:

1.3 Identifikation

Bat: Es folgt die Anzeige der Identifikationsseite von CODING1 (für CODING2/3 analog):

```
01111010101101101110000000001010001011011001000011110000110010101100101001000100
1                                                                              0
0                                                                              1
1                                                                              1
1                                                                              0
0                                                                              0
0                                                                              1
1                                                                              0
0                                                                              1
1                                                                              1
1                                                                              0
0    Programm zur direkten Ver-/Entschlüsselung von Dateien (1-Byte-Gruppen)   0
0                          von Jürgen Müller                                   0
0                                                                              0
1                        SYSJM Jürgen Müller                                   1
0                              6/2007                                          1
1                                                                              1
0                          Version 1.1                                         1
0                                                                              0
1      ---> n u r  f ü r  d e n  p r i v a t e n  G e b r a u c h <---          1
0                                                                              1
1  * Alle Rechte vorbehalten. Programm-Benutzung auf eigene Verantwortung.     0
1                                                                              1
11110011111011011010011011111010010111011001000011011011010110010011111100000010
```

1.4 Einschränkungen

Bat:/Dia: Bitte beachten Sie, dass die vorliegenden Versionen nicht zum kommerziellen Gebrauch berechtigen. Falls Sie eines dieser Programme für den kommerziellen Gebrauch verwenden wollen, wenden Sie sich bitte über eine der unten genannten Kontaktdatenangaben an den Autor. Vielen Dank.

Bitte beachten Sie darüber hinaus, dass trotz höchster Qualitätssicherheit für die Programme keine absolute Garantie übernommen werden kann. Falls wider Erwarten Probleme auftreten sollten, die nachweislich nicht durch falsche Bedienung verursacht werden, wird sich der Autor umgehend um eine Problem-Behebung bemühen. Natürlich sind auch jedwede Anregungen Ihrerseits zu den Programmen, Dokumentation usw. jeder Zeit willkommen.

Für das hier verwendete Verfahren (siehe Kurzexposé) gibt es **k e i n e r l e i** Möglichkeit, einen verlorenen Schlüssel wieder zu beschaffen, auch für den Autor nicht (!). Bitte arbeiten Sie im Zweifelsfall vorsichtshalber mit Schlüssel- bzw. Parameterdateien (siehe unten), um solche Probleme zu vermeiden. Vielen Dank für Ihr Verständnis.

CODING-Programme unter MS-Windows kennen **keine 4 GB-Beschränkung** für Dateigrößen (die aktuelle Grenze für Windows XP -Systeme liegt bei 9.223.372.036.854.775.807 Bytes).

Die „einfache“ und die „erweiterte“ Version eines CODING-Dialog-Programms sowie das zugehörige CODING-Batch-Programm sind voll kompatibel, die Programme CODING1, CODING2 und CODING3 untereinander jedoch nicht (!).

1.5 Allgemeines

Bat: Sie können das Programm nach dieser Eingangsmaske jeder Zeit während der Eingabe von Programm-Parametern entweder durch den Wert '9' bzw. ohne Eingabe weiterer Zeichen durch betätigen der RETURN(¿)-Taste ('Leereingabe') beenden, ohne dass eine Datei verschlüsselt oder entschlüsselt worden wäre.

Bei Überladeinformationen (siehe unten) können auch die Werte ’0’ und ’-1’ im Einzelfall dazu dienen.

Schließen Sie jede Eingabe wie gewohnt mit der RETURN(¿)-Taste ab.

Dia: Sie können jeder Zeit zu einzelnen Feldern Hilfetexte anfordern. Dies kann in Feldern, die eine Eingabe erfordern, mit der F1-Taste geschehen, für sonstige Felder ist das durch Doppel-Klick mit der linken Maustaste möglich.

Vor anklicken des „Starten“-Buttons finden keinerlei Aktionen 

des Programms bzgl. der Ver- bzw. Entschlüsselung von Dateien oder bzgl. der Bearbeitung von Parameterdateien statt. In dieser Phase bzw. nach der Bearbeitung kann das Programm jeder Zeit durch Betätigung des „Ende“-Buttons verlassen werden. Durch den „Rücksetzen“-Button können in diesen Phasen alle Felder wieder geleert bzw. auf ihre Standardwerte zurückgesetzt werden.

Bat:/Dia: Falls Sie relative Pfadangaben im Zusammenhang mit Dateinamen verwenden, berücksichtigen Sie, dass als Ausgangsverzeichnis das Verzeichnis angenommen wird, in dem die ausführbare Datei des angestarteten Programms bzw. die das Programm aufrufende Batchdatei (auch als Skript oder Batchprogramm bezeichnet) steht.

1.6 Kontrollausgabedatei

Bat: Nach kurzer Anzeige der Identifikationsseite wird die Startinformation des Programms ausgegeben und die Frage nach einer Datei für die Kontrollausgabe gestellt:

```
Ver-/Entschlüsselungsprogramm  C O D I N G 1  1.1  gestartet.
  Gestartet am: Do. 07.12.2006, 14:28:01.

Geben Sie bitte den Namen der Kontrollausgabedatei an, ggf. in runden
 Klammern ("(",")"), falls lediglich Fehler- und Dialog-Informationen
 auf den Bildschirm bzw. in diese Datei geschrieben werden sollen
  (Leereingabe = Programmabbruch):
```

Bat:/Dia: Geben Sie hier einen Dateinamen ggf. mit Pfad-Angabe für eine neu anzulegende oder zu überschreibende Datei an, in die alle relevanten Informationen (außer natürlich sicherheitsrelevante Informationen !) zu von Ihnen gewählten weiteren Parametern bzw. zu vom Programm erledigten Aufgaben protokolliert werden.

Dia: Um eine bestehende Datei und deren Pfadangabe in das Eingabefeld zu stellen, kann auf das links daneben stehende „Aktenmappe“-Symbol geklickt werden und nach Auswahl der Datei der „Öffnen“-Button im Dateiauswahl-Dialog betätigt werden. Danach können Sie diesen Namen im vorliegenden Eingabefeld noch beliebig editieren.

Bat:/Dia: Falls (**Bat:**) der Dateiname in Klammern gesetzt wird bzw. (**Dia:**) das Checkbox-Feld aktiviert wird (**Bat:/Dia:**), so werden keinerlei normale Informationen ausgegeben, weder in die Kontrollausgabe-Datei noch auf den Bildschirm. Lediglich im Fehlerfall werden dabei entsprechende Hinweise angezeigt und ausgegeben.

Das Programm arbeitet bei der Datei-Ausgabe mit dem ANSI-Zeichensatz (MS-Windows), nicht mit dem OEM-Zeichensatz (MS-DOS-Zeichensatz), was beim späteren Lesen der Kontrollausgabe-Datei zu berücksichtigen ist.

1.7 Ausgabe der Kommandozeilen-Steuerungs-Parameter

Bat: Nun können Sie sich eine detaillierte Beschreibung der Steuerungs-Parameter für den Kommandozeilen-Aufruf des Programms in diese Kontrolldatei ausgeben lassen:

```
Soll eine Parameter-Beschreibung des Programms
 in diese Kontrollausgabedatei ausgegeben werden ?
   0    - Nein, das Programm soll fortgesetzt werden,
   1    - Ja, danach soll das Programm fortgesetzt werden,
   2    - Ja, danach soll das Programm beendet werden.
Geben Sie bitte die zugehörige Zahl an:
```

Dies geschieht mit den Werten '1' bzw. '2', der Wert ’0’ übergeht diese Ausgabe. Für die Werte '0' bzw. '1' wird das Programm danach fortgesetzt.

Bitte beachten Sie, dass Ihnen die Frage nur dann gestellt wird, falls Sie zuvor nach dem Namen der Kontrollausgabedatei gefragt wurden. Wurde dieser als gültiger Name schon beim Aufruf des Programms als Kommandozeilen-Parameter mit angegeben, unterbleibt diese Abfrage und damit die Parameter-Ausgabe.

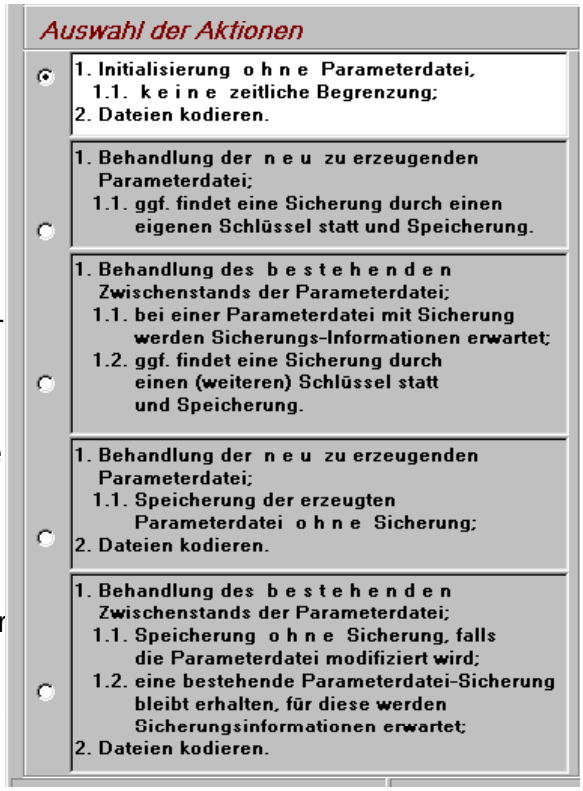
1.8 Parameterdatei-Bearbeitung

Bat: Haben Sie das Programm fortgesetzt, werden Sie gefragt, ob eine Parameterdatei-Bearbeitung stattfinden soll und damit ggf. eine zeitliche Begrenzung des Initialisierungsprozesses:

```
W A R N U N G: Die Initialisierung der Steuerparameter des Programmlaufs
               kann einige Zeit in Anspruch nehmen(!).
Sind Sie sicher,dass keine zeitliche Begrenzung des Initialisierungs-Prozesses
  vorgenommen, keine Zwischenergebnisse gebildet und in eine entsprechende
  Parameterdatei zwischengespeichert werden sollen ?
   0    - Nein, der Initialisierungs-Prozess wird zeitlich begrenzt,
          es findet eine Parameterdatei-Bearbeitung statt,
   1    - Ja, es gibt  k e i n e  zeitliche Begrenzung,
          es werden  k e i n e  Parameter-Zwischenergebnisse gespeichert,
   9    - Programmabbruch.
Geben Sie bitte die zugehörige Zahl an:
```

Bat:/Dia: Prinzipiell kann der Initialisierungsprozess speziell beim Programm CODING3 erhebliche Zeit in Anspruch nehmen. Auch bei den Programmen CODING1 und CODING2 kann dies dann der Fall sein, wenn Sie mit sehr großen Überlagerungsdateien und/oder einem sehr großen Iterationswert (siehe unten) arbeiten. Um diesen Zeitaufwand nicht bei jeder Ver- und Entschlüsselung in Kauf nehmen zu müssen, kann das Ergebnis des Initialisierungsprozesses in einer Datei – der Parameterdatei – gespeichert werden. Da sich der Initialisierungsprozess im Extremfall

(speziell Programm CODING3) über mehrere Stunden erstrecken kann, ist es möglich, Zwischenergebnisse des Initialisierungsprozesses abzuspeichern und später auf diesen Zwischenergebnissen wieder aufzusetzen, um den Initialisierungsprozess fortzuführen. Dies geschieht durch eine zeitliche Begrenzung des Initialisierungsprozesses.



Dia: In der „einfachen“ Version des Programms ist auf Grund der damit verbundenen wesentlich höheren Komplexität sowie des höheren Benutzer-Aufwands auf die Bearbeitung mit Parameterdateien verzichtet worden.

In der „erweiterten“ Version werden die im Batch-Programm einzeln abgefragten Eigenschaften durch die Auswahl der vorzunehmenden Arbeiten/Aktionen als ganzes zunächst festgelegt und damit bestimmte Parameterfelder, die benötigt werden, aktiviert bzw. Parameterfelder, die nicht benötigt werden, deaktiviert. Ggf. kann diese Festlegung durch andere Auswahlfelder auch wieder modifiziert werden.

Bat: Durch Eingabe des Wertes ’0’ geben Sie daher an, dass eine Parameterdatei verwendet bzw. verarbeitet werden soll und ggf. eine zeitliche Begrenzung berücksichtigt werden soll. Der Wert ’1’ ist daher nur sinnvoll, wenn Sie mit dem aktuellen Programmlauf eine Chiffrierung von Daten vornehmen wollen, ohne eine Parameterdatei einzusetzen.

1.8.1 Parameterdatei

Bat: Wurde eine Parameter-Behandlung gewünscht, so wird der Name der Parameterdatei erfragt:

```
Geben Sie bitte den Dateinamen der Parameterdatei an
  (Leereingabe = Programmabbruch):
```

Bat:/Dia: Bitte beachten Sie, dass sich die Größe einer Parameterdatei wie folgt berechnet:

*Parameterdatei-Größe := 512 + n * Bytes pro Byte-Gruppe * 2 ** (8 * Bytes pro Byte-Gruppe),*

mit n=3 für eine Zwischen- und n=2 für eine Ergebnis-Parameterdatei, d.h. 2.048 / 393.728 / 150.995.456 (n=3) bzw. 1.536 / 262.656 / 100.663.808 (n=2) Bytes für CODING1/2/3.

Eine im aktuellen Programmlauf verwendete Parameterdatei wird in diesem Programmlauf n i c h t behandelt, also auch nicht zerstört (siehe dazu 1.10. Ausnahmedatei unten).

Dia: Analog zur Beschreibung unter 1.6. Kontrollausgabedatei oben können Sie einen Dateinamen (mit Pfadangabe) in das Parameterdatei-Feld einstellen. Als Standard-Erweiterung wird dabei „par“ verwendet. Um Ihnen die Orientierung speziell auch bei längeren Namen zu erleichtern, wird Ihnen auf der linken Seite des Statusleisten-Feldes, das den unteren Abschluss der gesamten Eingabemaske bildet, angezeigt, wo Sie sich mit Ihrem Cursor befinden bzw. welche Zeichen Sie markiert haben. Solange das Parameterdatei-Feld nicht ausgewählt ist, d.h. nicht den Fokus besitzt, werden diese Informationen als Sterne angezeigt. Besitzt das Parameterdatei-Feld den Fokus, ist der Feldinhalt nur dann sichtbar, wenn sich der Maus-Zeiger im Kernbereich dieses Feldes befindet bzw. bewegt. Wird der Maus-Zeiger langsam aus dem Kernbereich heraus bewegt, verschwindet der Feldinhalt wieder (technisch: leider wird das adäquatere „OnMouseLeave“-Ereignis z.Zt. vom Entwicklungssystem nicht für solche Felder unterstützt). Unabhängig davon sind markierte Feldteile immer sichtbar.

1.8.2 Parameterdatei-Verwendung

Bat: Anschließend wird die Frage gestellt, ob die angegebene Parameterdatei neu anzulegen ist oder ob ihr Initialisierungszustand ggf. fortgeführt werden soll:

```
In welcher Form soll die vorhandene Parameterdatei verwendet werden ?
   0    - die Parameterdatei wird  n e u  angelegt,
   1    - es wird auf dem Zwischenzustand der Parameterdatei aufgesetzt,
   9    - Programmabbruch.
Geben Sie bitte die zugehörige Zahl an:
```

1.8.3 Chiffrierung nach Parameterdatei-Behandlung

Bat: Konnte die Parameterdatei erfolgreich geöffnet werden, so ist zu beantworten, ob nach einer ggf. vorzunehmenden Parameterdatei-Bearbeitung auch die Chiffrierung von Daten stattfinden soll:

```
Sollen nach der Steuerparameter-Bearbeitung Dateien kodiert werden ?
   0    - Nein, es soll nur eine Parameterdatei-Bearbeitung stattfinden,
   1    - Ja, es soll nach der Parameterdatei-Bearbeitung
               ggf. die Datei-Kodierung vorgenommen werden,
   9    - Programmabbruch.
Geben Sie bitte die zugehörige Zahl an:
```

Dia: In der „erweiterten“ Version wurden diese beiden Informationen schon mit der Auswahl der Aktion (siehe 1.8. oben) festgelegt.

1.8.4 Zeitgrenze

Bat: Falls der Parameterdatei-Zustand angibt, dass die Initialisierung noch nicht beendet ist, werden Sie nach einer Zeitgrenze gefragt, zu der Sie den Initialisierungsprozess entweder automatisch bzw. manuell beenden oder eine neue Zeitgrenze angeben können:

```
Bis ca. zu welchem Zeitpunkt soll der Steuerparameter-Initialisierungs-Prozess
  unterbrochen und das erzielte Zwischenergebnis in die Parameterdatei
  »parameterdatei« zwecks späterer Fortführung
  gespeichert werden ? (Leereingabe = Programmabbruch):
(in Form 'tt.mm.jjjj,ss:mm'; 99.99.9999,99:99 =unbegrenzt):
```

Haben Sie als Zeitgrenze „unbegrenzt“ angegeben, werden Sie aus Sicherheitsgründen nochmals zur Bestätigung Ihrer Wahl aufgefordert:

```
W A R N U N G: Die Initialisierung der Steuerparameter des Programmlaufs
               kann einige Zeit in Anspruch nehmen(!).
Sind Sie sicher,dass keine zeitliche Begrenzung des Initialisierungs-Prozesses
  vorgenommen werden soll ?
   0    - Nein, der Initialisierungs-Prozess wird zeitlich begrenzt,
   1    - Ja, es gibt  k e i n e  zeitliche Begrenzung,
   9    - Programmabbruch.
Geben Sie bitte die zugehörige Zahl an:
```

Dia: In der „erweiterten“ Version können Sie entweder unbegrenzt auswählen oder eine Zeitgrenze direkt eingeben. Beachten Sie bitte, dass das Programm Sie nochmals warnt, wenn zum Anstartzeitpunkt die von Ihnen gewählte Zeitgrenze schon in weniger als 10 Minuten erreicht wird.

1.8.5 Aktion bei Zeitgrenzen-Überscheitung

Bat: Wurde von Ihnen hingegen eine konkrete Zeitgrenze angegeben, so ist anzugeben, was zum Unterbrechungszeitpunkt geschehen soll:

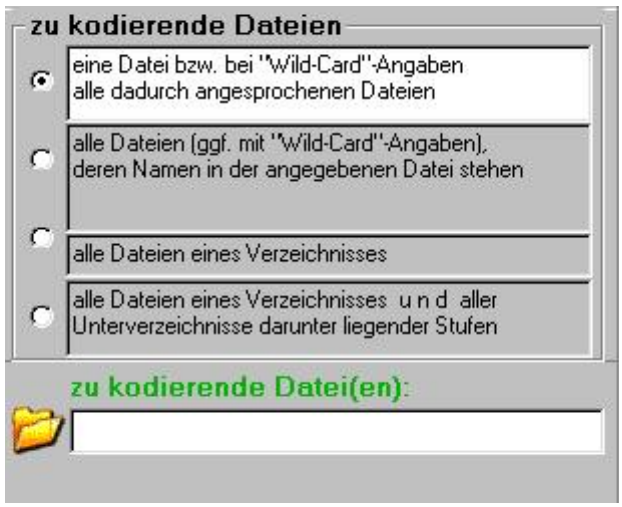
```
Das Programm soll nach Überschreitung der angegebenen Zeitgrenze
   0    - automatisch beendet werden,
   1    - ggf. mit neu nachgefragter Zeitgrenze fortgeführt werden.
   9    - Sofortiger Programmabbruch.
Geben Sie bitte die zugehörige Zahl an:
```

Damit ist die Eingabe der Parameterdatei-Behandlungsinformationen abgeschlossen. Nachfolgende Informationen zu den zu behandelnden Dateien, zu Ausnahmedateien, zur Bearbeitungsform und zur Statistik werden nur dann nachgefragt, wenn die oben gestellte Frage nach Chiffrierung von Daten positiv beantwortet wurde oder keine Parameterdatei-Bearbeitung gewünscht wurde.

1.9 Angaben zu den zu behandelnden Dateien

Bat: Soll nicht nur eine Parameterdatei-Behandlung stattfinden, werden Sie gefragt, in welcher Form Sie die Datei(en) angeben wollen, die verschlüsselt oder entschlüsselt werden soll(en):

```
Welche Dateien sollen bearbeitet werden?
   1  eine Datei bzw. bei "Wild-Card"-Angaben alle angesprochenen
      Dateien, deren Name nachfolgend angegeben wird
   2  alle Dateien (ggf. mit "Wild-Card"-Angaben), deren Namen in der
      nachfolgend angegebenen Datei stehen
   3  alle Dateien des nachfolgend angegebenen Verzeichnisses
   4  alle Dateien des nachfolgend angegebenen Verzeichnisses  u n d
      aller Unterverzeichnisse darunter liegender Stufen
   9  Programmabbruch.
Geben Sie bitte die zugehörige Zahl an:
```



Bat:/Dia: Durch (**Bat:**) Eingabe des Wertes ’1’ bzw. (**Dia:**) Auswahl der 1. Angabe (**Bat:/Dia:**) können Sie die Datei oder die Dateien, die nachfolgend behandelt werden sollen, ggf. mit Pfad-Angabe direkt angeben, wobei Sie die von MS-DOS gewohnten Sonderzeichen ’?’ für ein beliebiges Zeichen und ’*****’ für eine Gruppe beliebiger Zeichen im Dateinamen angeben können („Wild-Card-Angabe“), um mehrere Dateien auszuwählen.

Mit (**Bat:**) Eingabe des Wertes ’2’ bzw. (**Dia:**) Auswahl der 2. Angabe, die nur in der „erweiterten“ Version möglich ist, (**Bat:/Dia:**) teilen Sie dem Programm mit, dass Sie nachfolgend eine Datei angeben möchten, die selbst jeweils pro Dateizeile eine Dateiangabe oder eine Dateiangaben-Verzeichnis-Kombination für die Auswahl aller durch die Dateiangabe ausgewählten Dateien eines Verzeichnisses **und** dessen Unterverzeichnisse in spitzen Klammern („<“,“>“) beinhaltet, die zu behandelnde Dateien identifiziert. D.h. Sie haben zuvor eine Datei mit Dateinamen bzw. Dateiangaben der Dateien vorbereitet, die im vorliegenden Programmlauf behandelt werden sollen. Bitte beachten Sie hierbei, dass **Dateien, die von mehreren Dateiangaben angesprochen werden, auch mehrfach chiffriert werden !** Dies ist nur solange irrelevant, wie Sie diese Dateien nicht auch einzeln ansprechen wollen. Auf jeden Fall wird dadurch die Verarbeitungszeit entsprechend erhöht.

Dia: Haben Sie in der „erweiterten“ Version eine Dateinamen-Datei ausgewählt, können Sie deren Inhalt durch den „öffne Datei“-Button auch editieren. Dabei werden Ihnen schon während des Einlesens der Datei die Einträge angezeigt,die ggf. bzw. sicher ein oder mehrere Dateien doppelt ansprechen. Diese Einträge können Sie dann gezielt ausschließen. Weiter können Sie einzelne Dateien durch z.B. den Datei-Auswahldialog in das Dateinamen-Feld, das hier als Editierfeld dient, eintragen lassen und modifizieren, bevor Sie einen solchen Eintrag in die aufgeführte Liste übernehmen („einfügen“-Button). Der Eintrag wird über dem in der Liste markierten Eintrag eingefügt. Mit dem „löschen“-Button können Sie einen markierten Eintrag aus der Liste herausnehmen und in das Editierfeld übertragen, dort ggf. verändern und mit dem „einfügen“-Button wieder in die Liste einfügen.Bei jeder Eintragung in die bestehende Liste wird dabei vom Programm auf Überschneidung

mit einem bestehenden Listeneintrag geprüft. Falls Sie die Arbeiten beenden und die Liste wieder in die editierte Datei übertragen wollen, geschieht dies mit „speichere Liste in Datei“ und mit „schließe Datei“. Falls Sie Ihre Änderungen nicht übernehmen wollen, sollten Sie nach „schließe Datei“ die Sicherheits-Abfrage nach Speicherung der Liste verneinen. Als Standard erhalten solche Dateien die Erweiterung „dtn“.

Bat:/Dia: In vielen Fällen wünschen Sie, alle Dateien eines Verzeichnisses oder gar eines Verzeichnisses **u n d** sämtlicher Unterverzeichnisse beliebiger Stufe zu bearbeiten. Dies ist durch (**Bat:**) Angabe der Werte ’3’ bzw. (**Dia:**) Auswahl der 3. Angabe (**Bat:/Dia:**) für die Dateien genau eines Verzeichnisses und (**Bat:**) ’4’ bzw. (**Dia:**) Auswahl der 4. Angabe (**Bat:/Dia:**) für alle Dateien innerhalb eines Verzeichnisbaums möglich. In diesen Fällen wird Sie das Programm nachfolgend nach dem Verzeichnisnamen fragen, dem Sie natürlich eine Pfadangabe voranstellen können, wie Sie das auch von MS-DOS her kennen.

Dia: Durch klicken auf das „Aktenmappe“-Symbol starten Sie den Verzeichnis-Auswahl-Dialog, können ein Verzeichnis direkt auswählen und ggf. nach Auswahl im gezeigten Auswahlfeld noch editieren.

Bat: Je nach Eingabewert werden Sie daher wie folgt aufgefordert:

’1’:
```
Geben Sie bitte die Dateiidentifikation der zu bearbeitenden Datei(en) an
  (Leereingabe = Programmabbruch):
```
’2’:
```
Geben Sie bitte den Dateinamen der Datei an, in der die Namen
 der zu bearbeitenden Dateien enthalten sind
  (Leereingabe = Programmabbruch):
```
’3’, ’4’:
```
Geben Sie bitte den Namen des Verzeichnisses an, in dem die
 zu bearbeitenden Dateien aufgeführt sind
  (Leereingabe = Programmabbruch):
```

1.10 Ausnahmedatei

Bat: Danach werden Sie nach einer Ausnahmedatei gefragt:

```
Geben Sie bitte den Dateinamen der Datei an, in der die Namen
 der  n i c h t  zu bearbeitenden Dateien enthalten sind
 (Leereingabe = keine Ausnahmedatei):
```

Bat:/Dia: Diese Datei beinhaltet jeweils pro Dateizeile eine Dateiangabe oder eine Dateiangaben-Verzeichnis-Kombination, die Dateien identifiziert, die **n i c h t** zu behandeln sind, auch wenn sie zuvor als zu behandelnde Dateien angesprochen werden. D.h. Sie können zuvor eine Datei mit solchen Dateinamen bzw. Dateiangaben mit Pfadangaben vorbereiten, falls dies erforderlich ist.

Dia: Haben Sie in der „erweiterten“ Version eine Dateinamen-Datei ausgewählt, können Sie analog der Datei zu chiffrierender Dateien (s. 1.9. oben) deren Inhalt durch den „öffne Datei“-Button auch editieren bzw. neu erstellen. Als Standard erhalten solche Dateien die Erweiterung „exn“.

Bat:/Dia: Unabhängig von Ihrer Eingabe werden folgende Dateien im aktuellen Programmlauf **auf jeden Fall nicht bearbeitet:**

- die Moduldatei, die das aufgerufene Chiffrierungsprogramm beinhaltet,
- die Datei, die für die Kontrollausgabe verwendet wird,
- die verwendeten Dateien, die zu behandelnde Dateinamen beinhalten, falls solche angegeben werden (z.B. auch die hier angegebene Ausnahmedatei, falls eine solche angegeben wird),
- die verwendeten (Ausgangs-, Öffnungs- bzw. Abschluss-)Schlüsseldateien, falls angegeben,
- die verwendeten Überladedateien, falls solche angegeben werden,
- die verwendeten Parameterdateien, falls solche angegeben werden.

Falls Sie keine Ausnahmedatei angeben, geht das (aktuelle) Programm davon aus, dass alle übrigen von Ihnen zuvor angesprochenen Dateien auch tatsächlich bearbeitet werden sollen.

1.11 Bearbeitungsform

Bat:/Dia: Damit steht zwar fest, welche Dateien potentiell in die Bearbeitung mit einbezogen werden, aber es bleibt noch die Frage, in welcher Form dies geschehen soll: **Dia:**

Bat:

```
Wie soll(en) die Datei(en) bearbeitet werden?
   1  verschlüsseln der Datei(en) einzeln
   2  entschlüsseln der Datei(en) einzeln
   3  verschlüsseln der Datei(en) als  e i n e  Datenmenge
   4  entschlüsseln der Datei(en) als  e i n e  Datenmenge
   9  Programmabbruch.
Geben Sie bitte die zugehörige Zahl an:
```

Bat:/Dia: Die Begriffe „verschlüsseln“/“ chiffrieren“ und „entschlüsseln“/“dechiffrieren“ verstehen sich noch von selbst (obwohl diese Operationen auch vertauscht angewendet werden können !). Für die Sicherheit relevant bleibt aber die Frage, was z.B. unter verschlüsseln „als eine Datenmenge“ zu verstehen ist.

Zunächst können einzeln verschlüsselte Dateien auch einzeln, d.h. unabhängig von anderen Dateien wieder entschlüsselt werden. Dies ist bei der Verschlüsselung der Dateien als einer Datenmenge nicht möglich (!). Hier können nur genau die Dateien, die ursprünglich als eine Datenmenge verschlüsselt wurden, wieder gemeinsam entschlüsselt werden.

Der Vorteil der Verschlüsselung als einer Datenmenge ist, dass die Programm-intern verwendeten Schlüssel-Parameter nicht für jede Datei wieder auf den Ausgangszustand zurückgesetzt werden. Vielmehr werden alle beteiligten Dateien als eine große Datei behandelt, d.h. es findet eine Datei-übergreifende Datenblock-Bearbeitung statt, womit Entschlüsselungsversuche unberechtigter Dritter nochmals erheblich erschwert werden.

Innerhalb eines Verzeichnisses werden zunächst die Dateinamen des Verzeichnisses in alphabetischer Ordnung bearbeitet. Danach findet die Bearbeitung der Unterverzeichnisse, falls gewünscht, statt. Falls die Bearbeitung aller Dateien als eine Datenmenge angegeben wurde, werden auch die Unterverzeichnisse in ihrer alphabetischen Reihenfolge bearbeitet, sonst findet die Bearbeitung (aus Gründen der Vereinfachung) in der Reihenfolge statt, in der diese Verzeichnisse in internen Tabellen auftreten bzw. vom Betriebssystem zur Verfügung gestellt werden. Unabhängig davon werden bereits solche Verzeichnisbäume von der Verarbeitung ausgeschlossen, in denen gemäß einer Dateiangaben-Verzeichnis-Kombination der Ausnahmedatei keine zu verarbeitenden Dateien vorkommen können.

Als Nachteil der Verschlüsselung als einer Datenmenge bleibt, dass so verschlüsselte Verzeichnisse **n i c h t** modifiziert werden dürfen bzw. die Veränderungen müssen vor der Entschlüsselung exakt wieder in den Ausgangszustand überführt werden. Dies kann z.B. auch dadurch geschehen, dass man sämtliche bei der Verschlüsselung beteiligten Dateien in der Reihenfolge ihrer Verschlüsselung (siehe hierzu z.B. die Verschlüsselungs-Protokolldatei) in eine Datei einträgt und diese als Datei der zu behandelnden Dateien (siehe oben) für die Entschlüsselung angibt. Das vorliegende Programm arbeitet also nur auf Dateiinhalten, modifiziert also keine Unterverzeichnis-Informationen oder ähnliches.

Dies kann auch zur **weiteren Sicherung** verwendet werden. Falls Sie nach Verschlüsselung z.B. durch Umbenennung eine andere Datei an die 1. Stelle setzen, hilft auch kein korrekter Schlüssel!

1.12 Statistik

(**Bat:**) Mit der nachfolgenden Frage bzw. (**Dia:**) durch Auswahl der entsprechenden Check-Box (siehe oben) (**Bat:/Dia:**) beantworten Sie, ob Sie am Ende des Programmlaufs eine zusammen-fassende Statistik über das Chiffrierergebnis wünschen oder nicht. Eine solche Statistik zeigt Ihnen beim Verschlüsseln nochmals die Güte der konkreten Chiffrierung an, für den eigentlichen Chiffrierprozess spielt dies Angabe keinerlei Rolle.

Bat:

```
Soll eine Statistik über die Ergebnisdaten erstellt werden?
   0  nein, es wird  k e i n e  Ergebnisdaten-Statistik erstellt
   1  ja, es wird eine Ergebnisdaten-Statistik erstellt
   9  Programmabbruch.
Geben Sie bitte die zugehörige Zahl an:
```

1.13 Schlüssel-Informationen

Bat:/Dia:

Der Ablauf bis zur Chiffrierung von Daten ist bei allen CODING-Programmen identisch.

- Der Benutzer (Anwender) stellt dem Programm einen Ausgangs-Schlüssel zur Verfügung, der vom Benutzer direkt über die Tastatur (oder mit Hilfe von Maus-Bewegungen bei Parameterdatei-Behandlung) eingegeben oder über eine entsprechende Schlüsseldatei angegeben wird.
- Ggf. stellt der Benutzer ebenfalls sogenannte Überladeinformationen und entsprechende Überladeparameter zur Verfügung (siehe 2. unten).
- Das Programm berechnet aus diesen Ausgangsdaten einen Satz von Parametern, der zur eigentlichen Chiffrierung verwendet und ggf. in einer Parameterdatei gespeichert wird.
- Wird eine Parameterdatei verwendet, so kann diese Datei ebenfalls mit einem Parameterdatei-Sicherungsschlüssel gegen eine missbräuchliche Verwendung gesichert werden.

Der zu sichernde Teil einer Parameterdatei wird als ein gemeinsamer Datenblock aufgefasst und ggf. mit analogen Methoden, die sonst auch im Zusammenhang mit zu chiffrierenden Daten angewandt werden, gesichert. Für Parameterdateien kann als Ausgangs-Schlüssel ein beliebiger Zufallsschlüssel verwendet werden. Der Parameterdatei-Sicherungsschlüssel übernimmt in diesen Fällen die Sicherungsfunktion, die ursprünglich mit dem Ausgangs-Schlüssels verbunden war.

Der Teil der Parameterdatei, der ggf. chiffriert wird, ist intern so ausgelegt, dass er nach der Chiffrierung formal nicht von entsprechend unchiffrierten Parameterdatei-Teilen unterschieden werden kann. Speziell werden benötigte Transformationstabellen nur in Form ihrer Erzeugungsinformationen in Parameterdateien gehalten, die Tabellen selbst haben einen viel zu prägnanten Aufbau. Somit kann bei Entschlüsselungsversuchen von gesicherten Parameterdateien nicht auf Grund des dabei entstehenden Parameterdatei-Ergebnisses entschieden werden, ob der Entschlüsselungsversuch erfolgreich war. Auch „falsch“ entschlüsselte Parameterdateien sind funktionsfähig, die Programme können sie **nicht** von „korrekten“ Parameterdateien unterscheiden.

ACHTUNG: Falls Sie Parameterdateien mit Initialisierungs-Zwischenzuständen sichern, so bedenken Sie, dass diese Dateien sicherheitsrelevante Informationen beinhalten, auch wenn die endgültigen Chiffrierparameter darin noch nicht vollständig enthalten sind. Solche Dateien können z.B. Ihren Ausgangsschlüssel enthalten, der in vollständig initialisierten Parameterdateien nicht mehr enthalten ist und aus diesen auch nicht mehr hergeleitet werden kann (!). Zur Sicherheit sollten Sie solche Dateien **vor dem Löschen** mit einem „Zufalls“-Schlüssel verschlüsseln.

Die nun folgenden Informationen drehen sich sämtlich um alles, was mit Schlüsseln zu tun hat. Die gezeigten Meldungen entsprechen denen des Programms CODING1. Statt 256 bzw. 512 Zeichen werden bei CODING2 131.072 bzw. 262.144 Zeichen und bei CODING3 50.331.648 bzw. 100.663.296 Zeichen als maximale Zeichen angezeigt und potenziell akzeptiert.

Daher ist leicht einzusehen, dass bei CODING2 und erst recht bei CODING3 sinnvollerweise mit Schlüsseldateien bzw. mit Überladedateien im Direktmodus zu arbeiten ist, um nur ansatzweise die Bandbreite der möglichen Schlüssel ausschöpfen zu können.

Umfasst ein Schlüssel mehr als 256 /131.072 /50.331.648 Schlüsselzeichen, so wird der Rest der Zeichen ignoriert. Werden hingegen weniger als diese Anzahl Schlüsselzeichen angegeben, so werden die angegebenen Schlüsselzeichen so oft wiederholt und ggf. am Ende abgeschnitten, bis genau 256 /131.072 /50.331.648 Schlüsselzeichen für die weitere Verarbeitung vorliegen. Trotzdem sollten nicht zu wenige und nicht zu homogene Zeichen als Schlüsselzeichen angegeben

werden, da dies im Extremfall (z.B. nur hexadezimale Null-Zeichen) zu keiner Verschlüsselung führen könnte (!).

Bis auf diese wenigen Extremfälle ist dem Verschlüsselungsergebnis auch bei kurzen Schlüsseln nicht in irgend einer Form anzusehen, ob der Ursprungsschlüssel ein kurzer oder langer Schlüssel war. Lediglich beim Suchen nach dem Schlüssel wird ein Dritter wahrscheinlich zunächst kürzere Schlüssel ausprobieren, bevor er längere Schlüssel in Betracht zieht - und dann wohl aufgibt.

Prinzipiell werden alle Sonderzeichen außer Zeilenabschlusszeichen (meist durch RETURN(¿)-Taste erzeugt bzw. <carriage return><line feed> bzw. $0D_{hex}$– plus $0A_{hex}$– Zeichen) als Bestandteil eines Schlüssels aus ASCII-Zeichen aufgefasst. Zeilenabschlusszeichen dürfen auch bei der Schlüsseleingabe durch hexadezimale Zeichen beliebig verwendet werden, um z.B. Schlüsselinformationen auf einzelne Zeilen aufzuteilen. (**Bat:**) Zeilenabschlusszeichen werden bei der Schlüsseleingabe über Tastatur allgemein ignoriert bzw. in Form einer zusätzlichen Leerzeile als Schlüsselende interpretiert. (**Bat:/Dia:**) In Schlüsseldateien mit ASCII-Zeichen-Schlüssel werden jedoch auch Zeilenabschlusszeichen als Teil des Schlüssels aufgefasst.

1.13.1 Abfrage zur Parameterdatei-Sicherung

Bat: Falls eine Parameterdatei Verwendung findet, die Initialisierung der Parameterdatei abgeschlossen ist und **keine Chiffrierung** von Daten vorgenommen werden soll, so werden Sie zur Sicherung dieser Parameterdatei befragt:

```
Soll die Parameterdatei mit einem Sicherungsschlüssel gesichert werden?
  '0'   - Nein, keine Parameterdatei-Sicherung,
  'S'   - Ja, Sicherung der Parameterdatei mit einem eigenen Schlüssel.
  '9'   - Sofortiger Programmabbruch.
Geben Sie bitte die zugehörige Zahl bzw. Zeichen an:
```

Falls Sie die Sicherung mit einem Schlüssel wünschen, werden Sie im folgenden zu den Angaben des Parameterdatei-Abschluss-Sicherungsschlüssels befragt.

Bat:/Dia: Bevor Sie zukünftig diese Parameterdatei verwenden können, wird Sie das Programm zunächst nach diesen Schlüsselangaben als Parameterdatei-Öffnungs-Sicherungsschlüssel befragen, soweit Sie diesen Sicherungsschlüssel nicht durch verneinen oben gestellter Frage bei einem erneuten Programmlauf ohne Daten-Chiffrierung mit der Parameterdatei entfernen lassen.

Im Folgenden gelten die Angaben, die sich auf „Ausgangs-Schlüssel“ beziehen, auch analog für „Parameterdatei-Öffnungs- und -Abschluss-Sicherungsschlüssel“, falls nichts anderes erwähnt ist.

Dia: Wie schon unter 1.8. Parameterdatei-Bearbeitung erwähnt, kann in der „einfachen“ Version des Programms nur mit Ausgangs-Schlüsselinformationen gearbeitet werden, auch ein Zufallsschlüssels ist dabei nicht sinnvoll.

Durch die Auswahl der Aktion (siehe 1.8.) in der „erweiterten“ Version wird bereits festgelegt, welche der gezeigten Schlüsselinformationen tatsächlich benötigt werden, und nur diese können daher von Ihnen eingegeben werden.

1.13.2 Zeichensatz

Bat:

```
Mit welchem Zeichensatz möchten Sie den Ausgangs-Schlüssel eingeben?
   1  ASCII-Zeichen (max. 256 Zeichen)
   2  hexadezimale Zeichen
      (max. 512 Zeichen "0" bis "9" bzw. "A" bis "F")
   9  Programmabbruch.
Geben Sie bitte die zugehörige Zahl an:
```

Bat:/Dia: Hierbei geht es um den Zeichensatz, mit dem Schlüsselinformationen eingegeben bzw. eingelesen werden sollen. Um alle möglichen Zeichen an- und eingebbar zu machen, kann neben dem „normalen“ Zeichensatz (ASCII-Zeichen) für Zeichen, die z.B. durch die Tastatur eingegeben werden können, auch ein Zeichensatz gewählt werden (hexadezimale Zeichen), bei dem jeweils zwei Zeichen vom Programm zu einem Schlüsselzeichen zusammengesetzt werden. Damit lassen sich auch Schlüsselzeichen erzeugen, die mit der Tastatur (d.h. direkt) ggf. nicht oder nur sehr mühsam erzeugbar sind.

Dia: Mit dem Checkbox-Feld rechts über dem Schlüsselfeld können Sie entscheiden, ob Sie Schlüsselinformationen direkt bzw. aus einer Datei in hexadezimaler Form oder als ASCII-Zeichen eingeben bzw. einlesen wollen. Im Dialog haben Sie die Möglichkeit, auch während der Eingabe des Direkt-Schlüssels jeder Zeit vom normalen Zeichensatz der Tastatur auf den hexadezimalen Zeichensatz und umgekehrt umzuschalten. Dabei werden ASCII-Zeichen automatisch in ihre hexadezimalen Werte expandiert bzw. in umgekehrter Richtung werden hexadezimale Werte auf ASCII-Zeichen verdichtet. Letztere Möglichkeit besteht jedoch nur dann, wenn keine Zeichenwerte 00_{hex} im Schlüssel vorkommen (die in der Rechner-internen Darstellung als Zeichenketten-Abschluss interpretiert werden).

Des weiteren kann es bei der Schlüsseleingabe beim Verlassen eines Schlüsselfeldes durch einen Maus-Klick vereinzelt zu Problemen kommen, die vermieden werden können, wenn Sie einen Direkt-Schlüssel durch die Return-Taste abschließen. Trotz entsprechenden Bemühungen des Autors konnten hier bis jetzt nicht alle Unebenheiten beseitigt werden.

1.13.3 Eingabe-Modus

Bat:

```
In welchem Eingabe-Modus möchten Sie den Ausgangs-Schlüssel eingeben?
   0  zweimal über Tastatur  o h n e  Echo auf dem Bildschirm
       (das 2. Mal zur Kontrolle!)
   1  einmal über Tastatur  m i t  Bildschirm-Kontrolle
   2  über eine Datei, die den Schlüssel enthält
   9  Programmabbruch.
Geben Sie bitte die zugehörige Zahl an:
```

Dia: Im Auswahlbereich über dem Schlüsselfeld (siehe oben) können Sie sich zwischen der direkten Eingabe des Schlüssels über das Schlüsselfeld und der Eingabe durch eine Schlüsseldatei entscheiden. In der „erweiterten“ Version haben Sie bei Bearbeitung mit Hilfe einer Parameterdatei noch die Möglichkeit, einen Zufallsschlüssel mit Hilfe von Maus-Bewegungen zu erzeugen bzw. die Parameterdatei unverschlüsselt zu behandeln.

1.13.4 Tastatur-Eingabe

Bat: Falls Sie die Schlüsselzeichen über die Tastatur eingeben wollen, wird Ihnen mit (Modus=’1’):

```
Geben Sie bitte Ihren Ausgangs-Schlüssel in Hexadezimal-/ASCII-Zeichen
 (Schlüssel-Ende: Leerzeile) ein:
```

bzw. mit (Modus=’0’):

```
Geben Sie bitte Ihren Ausgangs-Schlüssel in Hexadezimal-/ASCII-Zeichen
 (Schlüssel-Ende: Leerzeile) das 1. Mal ein:
```

und mit:

```
Geben Sie bitte Ihren Ausgangs-Schlüssel in Hexadezimal-/ASCII-Zeichen
 (Schlüssel-Ende: Leerzeile) das 2. Mal ein (Kontroll-Eingabe):
```

die Gelegenheit gegeben.

Dia: Bei der Eingabe des Direktschlüssels sehen Sie links neben dem Schlüsselfeld zunächst ein „Lupe“-Symbol  . Dieses Symbol zeigt an, dass der entsprechende Schlüssel noch nicht eingegeben bzw. noch nicht durch Wiederholung des Schlüssels bestätigt wurde. Jedes Zeichen in diesem Feld wird durch ein Stern-Zeichen dargestellt. Es gibt keine Möglichkeit, Schlüsselzeichen des Feldes direkt zu sehen. Um Ihnen die Orientierung speziell auch bei längeren Schlüsseln zu erleichtern, wird Ihnen auch hier im Statusleisten-Feld angezeigt, wo Sie sich mit Ihrem Cursor befinden bzw. welche Zeichen Sie markiert haben.

Haben Sie Ihren Schlüssel vollständig eingegeben (siehe auch die Bemerkungen zu 1.13.2. oben), können Sie dies dem Programm durch die Return-Taste oder durch einen Maus-Klick außerhalb des Schlüsselfeldes und des Checkbox-Feldes für die hexadezimale Eingabe mitteilen. Nun wird das Programm fragen, ob Sie den Schlüssel wiederholen wollen. Falls Sie mit „nein“ antworten, ist die Schlüsseleingabe beendet und das „Lupe“-Symbol bleibt als Erinnerung erhalten. Sie können jeder Zeit durch anklicken dieses „Lupe“-Symbols die Bestätigung des Schlüssels nachholen. Haben Sie „ja“ geantwortet, erhalten Sie unter der Feldüberschrift „Schlüssel-Wiederholung“ die Gelegenheit, den Schlüssel durch Wiederholung zu bestätigen. Auch hier können Sie dem Programm durch die Return-Taste oder durch einen Maus-Klick außerhalb des Schlüsselfeldes und des Checkbox-Feldes für die hexadezimale Eingabe den Abschuss der Eingabe mitteilen.

Stimmen beide Schlüssel-Eingaben überein, erscheint ein „Häkchen in grünem Kreis“-Symbol  , das Ihnen anzeigt, dass diese Schlüsseleingabe erfolgreich abgeschlossen wurde. Jede weitere Änderung des Schlüssels wird vom Programm als Neueingabe des Schlüssels gewertet und erneut wie beschrieben behandelt.

1.13.5 Eingabe via Schlüssel-Datei

Bat: Falls Sie eine Schlüsseldatei verwenden wollen (Modus=’2’), werden Sie mit:

```
Wie möchten Sie den Namen der Datei eingeben, die den Ausgangs-Schlüssel
                                                              enthält?
   0  zweimal über Tastatur  o h n e  Echo auf dem Bildschirm
       (das 2. Mal zur Kontrolle!)
   1  einmal über Tastatur  m i t  Bildschirm-Kontrolle
   9  Programmabbruch.
Geben Sie bitte die zugehörige Zahl an:
```

zunächst nach der Art der Eingabe des Dateinamens der Schlüsseldatei gefragt.

Danach wird mit:

```
Geben Sie bitte den Namen der Datei, die den Ausgangs-Schlüssel enthält,
 an (Leereingabe = Programmabbruch):
```

bzw. mit:

```
Geben Sie bitte den Namen der Datei, die den Ausgangs-Schlüssel enthält,
 das 1. Mal an (Leereingabe = Programmabbruch):
```

und mit:

```
Geben Sie bitte den Namen der Datei, die den Ausgangs-Schlüssel enthält,
 das 2. Mal an (Kontroll-Eingabe):
```

der Dateiname erfragt.

Dia: Haben Sie im Auswahlbereich für die Schlüsseleingabe den Eintrag „Schlüsseldatei“ ausgewählt, so erscheint das „Aktenmappe“-Symbol links neben dem Schlüsseleingabefeld. Analog zur Beschreibung unter 1.6. Kontrollausgabedatei oben können Sie einen Dateinamen (mit Pfadangabe) in das Schlüsselfeld einstellen. Auch hier wird Ihnen in der Statusleiste angezeigt, wo sich Ihr Cursor befindet. Solange das Schlüsselfeld nicht ausgewählt ist, d.h. nicht den Fokus besitzt, werden diese Schlüsselfeld-Informationen als Sterne angezeigt. Besitzt das Schlüsselfeld den Fokus, ist der Feldinhalt nur dann sichtbar, wenn sich der Maus-Zeiger im Kernbereich des Schlüsselfeldes befindet bzw. bewegt. Wird der Maus-Zeiger langsam aus dem Kernbereich heraus bewegt, verschwindet der Feldinhalt wieder. Unabhängig davon sind markierte Feldteile immer sichtbar.

Bat:/Dia: Verwenden Sie eine Schlüsseldatei, wenn Sie einen komplexen Schlüssel verwenden wollen oder müssen. Eine im aktuellen Programmlauf verwendete Schlüsseldatei wird in diesem Programmlauf n i c h t behandelt (also auch nicht zerstört), selbst wenn diese explizit oder implizit als zu behandelnde Datei angegeben wurde (siehe dazu 1.10. Ausnahmedatei oben).

1.13.6 Zufallsschlüssel-Eingabe

Dia: Bei der Zufallsschlüssel-Eingabe im Zusammenhang mit der Behandlung von Parameterdateien gibt es die Möglichkeit, einen durch Maus-Bewegungen und Tastendrücke zufällig erzeugten Schlüssel als Ausgangsschlüssel für die Erzeugung einer Parameterdatei zu generieren. Durch klicken auf das „Maus“-Symbol  erscheint ein Hinweisfenster, das Ihnen die konkrete Handhabung dazu erklärt, bevor Sie nach Bestätigung mit der Schlüsselerzeugung beginnen können. Während des Schlüsselgenerierungsprozesses erscheint der Text „>> Zufallsschlüssel << -Erzeugung ...“ im Schlüsselfeld und ein Zähler in der Statusleiste zeigt Ihnen an, wie viele Elemente Sie schon erzeugt haben. Sind alle Elemente eines Zufallsschlüssels erzeugt oder wird der Erzeugungsprozess vorzeitig beendet, so werden Sie in einem separaten Dialogfenster darauf hingewiesen.

War die Schlüsselerzeugung erfolgreich, erscheint das „Häkchen in grünem Kreis“-Symbol. Falls Sie einen Zufallsschlüssel verwerfen wollen, kann das durch anklicken dieses Symbols geschehen.

2 Beschreibung der Überladeinformationen von CODING

Bat:/Dia: Die nachfolgenden Angaben beziehen sich auf sogenannte „Überladeinformationen“ (auch „Überlagerungsinformationen“), die speziell für die Programme CODING2 und CODING3 von wesentlicher Bedeutung sind. Die Werte der angezeigten Meldungen beziehen sich auf CODING1 und sind für CODING2 und CODING3 analog zu modifizieren.

Unter „Überladen“ wird die Möglichkeit verstanden, einen Schlüssel durch gezielte Veränderungen manuell oder auch durch Dateiangaben zu modifizieren, was zu einem sogenannten abgeleiteten Schlüssel führt. Die Veränderungen können dabei auf bestimmte Stellen des Schlüssels, auf bestimmte Schlüsselwerte und sie können auf bestimmte Änderungswerte innerhalb der Überladeinformationen beschränkt werden.

Damit ist es z.B. im Direktmodus möglich, eine zweite Datei, die Überladedatei, dazu zu verwenden, Werte an „zufälligen“ Stellen im bisherigen Schlüssel durch gewisse „zufällige“ Werte, die aus der Überladedatei abgeleitet werden, zu überschreiben.

Dia: Überladeinformationen werden nur dann verwendet, wenn das entsprechende Checkbox-Feld „mit Überlade-Informationen“ unter dem Ausgangs-Schlüsselfeld gesetzt wird (siehe 1.13.1 oben) oder eine Parameterdatei (s. 1.8 oben) im Zustand „noch(mals) zu überladen“ Verwendung findet.

2.1 Überladeinformation

Bat:/Dia: Allgemein besteht jede einzelne Überladeinformation aus zwei Angaben:

- der Stellenangabe und
- dem Wert, aus dem der Überladewert abgeleitet wird.

Im „Direktmodus“ bestehen diese Angaben jeweils aus den Zeichenwerten hintereinander liegender Zeichen innerhalb einer beliebigen Datei bzw. innerhalb einer beliebigen Eingabezeichenkette (Wert ’7’ unten). Je nach Programm (CODING1/2/3) ermitteln sich die Werte für die Stellenangaben und die Überladewerte dabei jeweils aus 1 bzw. 2 bzw. 3 Zeichen.

Bat: Falls nicht der Direktmodus vorliegt, müssen diese Angaben wie folgt aussehen:

```
Sollen Schlüsselangaben überladen werden?
   0  nein, keine Schlüsselangaben-Überladung
   1  ja, in der Form "<bytenr-dez>:<zeichen>,"
   2  ja, in der Form "<bytenr-dez>:<hexwert>,"
   3  ja, in der Form "<bytenr-dez>:<dezwert>,"
   4  ja, in der Form "<bytenr-hex>:<zeichen>,"
   5  ja, in der Form "<bytenr-hex>:<hexwert>,"
   6  ja, in der Form "<bytenr-hex>:<dezwert>,"
   7  ja, in Form einer fortlaufenden Zeichenkette
   9  Programmabbruch.
Geben Sie bitte die zugehörige Zahl an:
```

Dabei steht

’bytenr-dez/-hex’	für die dezimale bzw. hexadezimale Nummer der angesprochenen Schlüsselstelle zwischen 1 und 256 /65.536 /16.777.216 bzw. zwischen 1 und 100_{hex} /10000_{hex} /1000000_{hex},
’zeichen’	für 1 /2 /3 beliebige(s) Zeichen,
’hexwert’	für hexadezimale Halbbytezeichen aus 0 bis 9 bzw. A bis F mit einem Wert zwischen 0 und FF_{hex} /$FFFF_{hex}$ /$FFFFFF_{hex}$,
’dezwert’	für einen Dezimalwert zwischen 0 und 255 /65.535 /16.777.215.

Dia: In der „einfachen“ Version des Programms können Überladeinformationen nur im Direktmodus aus Überladedateien übernommen werden. In der „erweiterten“ Version lässt sich die konkrete Metasytaxform bzw. der Direktmodus auswählen, die Eingabe der Überladeinformationen ist hier auch direkt möglich.

Auch die Intervalle zu den Schlüssel-Überladestellen, den Schlüssel-Überladewerten sowie zu den zu überladenden Schlüsselwerten können nur in der „erweiterten“ Version vom Benutzer eingeschränkt werden, in der „einfachen“ Version wird der Standard „alle Werte“ verwendet.

Aus Sicherheitsgründen gilt auch für diese Intervallfelder, dass solange ein Feld nicht ausgewählt ist, d.h. nicht den Fokus besitzt, werden die Informationen des Feldes als Sterne angezeigt. Besitzt das Feld den Fokus, ist der Feldinhalt nur dann sichtbar, wenn sich der Maus-Zeiger im Kernbereich dieses Feldes befindet bzw. bewegt. Wird der Maus-Zeiger langsam aus dem Kernbereich heraus bewegt, verschwindet der Feldinhalt wieder.

Bat:/Dia: Wird mit einer Parameterdatei gearbeitet und soll keine Chiffrierung von Daten vorgenommen werden, so ist es möglich, den Schlüssel der Parameterdatei für weitere Überladungen in späteren Programmläufen vorzusehen. Dazu wird die Parameterdatei nach Bearbeitung der hier vorliegenden Überladung im Zustand „noch(mals) zu überladen“ gespeichert und das Programm beendet.

Dia: In der „erweiterten“ Version kann dies durch Auswahl des Checkbox-Feldes „nach Überladen speichern“ geschehen. **Bat:** Um zu erfahren, ob Sie dies wünschen, wird folgende Frage gestellt:

```
Soll der Schlüssel in einem späteren Programmlauf noch(mals) überladen werden?
   0  nein, die Schlüsselüberladung ist mit dem aktuellen Lauf abgeschlossen,
   1  ja, es soll ggf. eine weitere Schlüsselüberladung stattfinden
      und das Programm wird nach der jetzigen Überladung beendet.
   9  Programmabbruch.
Geben Sie bitte die zugehörige Zahl an:
```

Bat:/Dia: Damit ist es möglich, „beliebig“ komplexe Überladungen automatisiert durchzuführen.

2.2 Stellen-Intervall

Bat:

```
Geben Sie bitte das Intervall zu berücksichtigender Stellen
 im Schlüssel an, die überladen werden dürfen:
  kleinste mögliche Stellenangabe (1 bis 256 (0=Abbruch)):
```

und

```
  größte mögliche Stellenangabe (nnn bis 256 (0=Abbruch)):
```

Bat:/Dia: Durch diese Angaben wird festgelegt, auf welche Stellen sich Überladeinformationen beziehen müssen, um für die Änderung des Schlüssels potentiell von Bedeutung zu sein. Erst wenn zusätzlich die nachfolgend beschriebenen Überladewerte- und Schlüsselwerte-Intervall-Angaben ebenfalls erfüllt sind, führt eine Überladeinformation tatsächlich zum Überladen einer Schlüsselstelle mit dem in der Überladeinformation angegebenen Wert.

Änderungen finden prinzipiell nur an den Stellen innerhalb des Stellen-Intervalls statt.

2.3 Überladewerte-Intervall

Bat:

```
Geben Sie bitte das Intervall zu berücksichtigender Werte an, mit denen
 die Stellen im Schlüssel überladen werden dürfen (in Dezimal-Darstellung):
  kleinste mögliche Wertangabe (0 bis 255 (-1=Abbruch)):
```

und

```
  größte mögliche Wertangabe (nnn bis 255 (-1=Abbruch)):
```

Bat:/Dia: Durch diese Angaben wird festgelegt, welche Wertangaben Überladeinformationen beinhalten müssen, um für die Änderung des Schlüssels potentiell von Bedeutung zu sein. Erst wenn zusätzlich die zuvor beschriebene Stellen-Intervall-Angabe und die nachfolgend beschriebene Schlüsselwerte-Intervall-Angabe ebenfalls erfüllt sind, führt eine Überladeinformation tatsächlich zum Überladen einer Schlüsselstelle mit dem in der Überladeinformation angegebenen Wert.

2.4 Schlüsselwerte-Intervall

Bat:

```
Geben Sie bitte das Intervall zu berücksichtigender Schlüsselwerte an,
 die überladen werden dürfen (in Dezimal-Darstellung):
  kleinste mögliche Schlüsselwertangabe (0 bis 255 (-1=Abbruch)):
```

und

```
  größte mögliche Schlüsselwertangabe (nnn bis 255 (-1=Abbruch)):
```

Bat:/Dia: Durch diese Angaben wird festgelegt, welche Wertangaben Schlüsselinformationen beinhalten müssen, um für die Überladung potentiell in Frage zu kommen. Erst wenn zusätzlich die zuvor beschriebenen Stellen- und Überladewerte-Intervall-Angaben ebenfalls erfüllt sind, führt eine Überladeinformation tatsächlich zum Überladen einer Schlüsselstelle mit dem in der Überladeinformation angegebenen Wert.

Allgemein kann eine Stelle im Schlüssel auch mehrfach überladen werden. Nach einer tatsächlich vorgenommenen Überladung wird der Überladewert als neuer Schlüsselwert an der Überladestelle für die weitere Überlade-Verarbeitung angesehen. Die letzte auf eine Stelle anzuwendende Überladeinformation entscheidet, welcher Wert der endgültige Schlüsselwert an dieser Stelle sein wird.

Bei der Überladung wird der eingegebene bzw. aus einer Datei stammende Wert zuvor gemäß

CODING1:

*Überladewert := (Eingabe-/Dateiwert + bisheriger Schlüsselwert ***** 113*
+ Schlüsselwert an vorheriger Stelle bzw. Stelle 256 für Stelle 1) modulo 256

CODING2:

*Überladewert := (Eingabe-/Dateiwert + bisheriger Schlüsselwert ***** 30.781*
+ Schlüsselwert an vorheriger Stelle bzw. Stelle 65.536 *für Stelle 1) modulo* 65.536

CODING3:

*Überladewert := (Eingabe-/Dateiwert + bisheriger Schlüsselwert ***** 100.000.000.019*
+Schlüsselwert an vorheriger Stelle bzw. Stelle 16.777.216 *für Stelle 1) modulo* 16.777.216

transformiert, bevor der daraus hergeleitete Wert als eigentlicher Überladewert Anwendung findet. Stellen-, Überlade- und Schlüsselwerte-Intervall-Angaben sind hauptsächlich für Überladedateien im Direktmodus von Interesse. Im Direktmodus kann **jede** Datei als Überladedatei angegeben werden, auch Programm-Dateien (*****.COM, *****.EXE) bzw. sonstige Binärdateien (**!**).

Allgemein ist zu beachten, dass für alle Stellen innerhalb des Stellen-Intervalls **im Zusammenhang mit einer beliebig variierenden, (fast) beliebig langen Überladeinformation** gilt:

Schlüsselwerte aus dem Schlüsselwerte-Intervall werden am Ende (fast) vollständig in Überladewerte des Überladewerte-Intervalls überführt.

Falls es Überladewerte gibt, die nicht im Schlüsselwerte-Intervall liegen, so gilt **noch einschränkender**:

Alle Schlüsselwerte werden am Ende (fast) vollständig in Werte des Überladewerte-Intervalls, die nicht im Schlüsselwerte-Intervall vorkommen, überführt (!).

Je besser eine Überladeinformation die genannten Bedingungen erfüllt, desto mehr gelten genannte Folgerungen, was unbedingt zu beachten ist, um Trivial-Schlüssel zu vermeiden !

Ist ein Ausgangsschlüssel, eine Überladeinformation und ein Überladewerte-Intervall gegeben, so **gilt allgemein**:

Schlüsselwerte-Intervalle, die dieses Überladewerte-Intervall umfassen, liefern den gleichen Ergebnisschlüssel, falls für alle Schlüsselwerte, die im Schlüsselwerte-Intervall aber nicht im Überladewerte-Intervall enthalten sind, gilt, dass sie nicht im Ausgangsschlüssel vorkommen.

2.5 Eingabe-Modus

Bat:

```
In welchem Eingabe-Modus möchten Sie die Überladeinformationen eingeben?
   0  zweimal über Tastatur  o h n e  Echo auf dem Bildschirm
       (das 2. Mal zur Kontrolle!)
   1  einmal über die Tastatur  m i t  Bildschirm-Kontrolle
```

für Überladeinformationen der Informationsart 1 bis 6:

```
   2  über eine Datei, die die Überladeinformation enthält
```

für Überladeinformationen der Informationsart 7:

```
   2  über eine Datei bzw. bei "Wild-Card"-Angaben über alle angesprochenen
      Dateien, deren Name nachfolgend angegeben wird
   3  über alle Dateien (ggf. mit "Wild-Card"-Angaben), deren Namen in der
      nachfolgend angegebenen Datei stehen
   4  über alle Dateien des nachfolgend angegebenen Verzeichnisses
   5  über alle Dateien des nachfolgend angegebenen Verzeichnisses  u n d
      aller Unterverzeichnisse darunter liegender Stufen
```

und:

```
   9  Programmabbruch.
Geben Sie bitte die zugehörige Zahl an:
```

Dia: In der „einfachen“ Version des Programms können Überladeinformationen nur aus Überladedateien übernommen werden, in der „erweiterten“ Version können diese auch direkt angegeben werden.

2.6 Tastatur-Eingabe

Bat: Falls Sie die Überladeinformationen über die Tastatur eingeben wollen, wird Ihnen je nach Informationsart mit:

’1’:

```
Geben Sie bitte Ihre Überladeinformation in der Form "<bytenr-dez>:<zeichen>,"
```

’2’:

```
Geben Sie bitte Ihre Überladeinformation in der Form "<bytenr-dez>:<hexwert>,"
```

’3’:

```
Geben Sie bitte Ihre Überladeinformation in der Form "<bytenr-dez>:<dezwert>,"
```

’4’:

```
Geben Sie bitte Ihre Überladeinformation in der Form "<bytenr-hex>:<zeichen>,"
```

’5’:

```
Geben Sie bitte Ihre Überladeinformation in der Form "<bytenr-hex>:<hexwert>,"
```

’6’:

```
Geben Sie bitte Ihre Überladeinformation in der Form "<bytenr-hex>:<dezwert>,"
```

’7’:

```
Geben Sie bitte Ihre Überladeinformation in Form einer fortlaufenden
                                                              Zeichenkette
```

jeweils mit

```
   (Info-Ende: Leerzeile) ein:
```

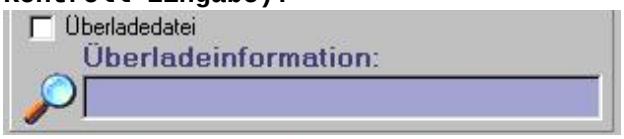
bzw.

```
   (Info-Ende: Leerzeile) das 1. Mal ein:
```

bzw.

```
   (Info-Ende: Leerzeile) das 2. Mal ein (Kontroll-Eingabe):
```

die Gelegenheit gegeben.



Dia: In der „erweiterten“ Version können Sie die Überladeinformationen im entsprechenden Feld direkt eingeben. Falls die Überladeinformation im Direktmodus eingegeben wird, so werden diese Informationen analog wie Direktschlüssel-Informationen einschließlich Wiederholung (siehe 1.13.4. oben) behandelt. Sind diese Informationen in Metasyntax bereit zu stellen, so gilt, dass solange ein Feld nicht ausgewählt ist, d.h. nicht den Fokus besitzt, die Informationen des Feldes als Sterne angezeigt werden. Besitzt das Feld den Fokus, ist der Feldinhalt nur dann sichtbar, wenn sich der Maus-Zeiger im Kernbereich dieses Feldes befindet bzw. bewegt. Wird der Maus-Zeiger langsam aus dem Kernbereich heraus bewegt, verschwindet der Feldinhalt wieder. Unabhängig davon sind markierte Feldteile immer sichtbar. Das „Lupe“-Symbol dient in diesen Fällen dazu anzuzeigen, dass die Syntax der Überladeinformation bzgl. der gewählten Metasyntax noch nicht erfolgreich geprüft wurde.

2.7 Eingabe via Überlade-Dateiangaben

Bat: Falls Sie Überlade-Dateiangaben verwenden wollen, werden Sie mit:

```
Wie möchten Sie die Überlade-Dateiangabe eingeben?
   0  zweimal über Tastatur  o h n e  Echo auf dem Bildschirm
       (das 2. Mal zur Kontrolle!)
   1  einmal über Tastatur  m i t  Bildschirm-Kontrolle
   9  Programmabbruch.
Geben Sie bitte die zugehörige Zahl an:
```

zunächst nach der Art der Eingabe der Dateiangabe gefragt. Danach wird

für die Informationsarten 1 bis 6:

```
Geben Sie bitte den Namen der Datei, die die Überladeinformation enthält,
```

für Informationsart 7, Eingabe-Modus '2':

```
Geben Sie bitte die Dateiidentifikation der Überladedatei(en)
```

für Informationsart 7, Eingabe-Modus '3':

```
Geben Sie bitte den Dateinamen der Datei an, in der die Namen
 der Überladedateien enthalten sind
```

für Informationsart 7, Eingabe-Modus '4', '5':

```
Geben Sie bitte den Namen des Verzeichnisses an, in dem die
 zu Überladedateien aufgeführt sind
```

alle mit:

```
an (Leereingabe = Programmabbruch):
```

bzw. mit:

```
das 1. Mal an (Leereingabe = Programmabbruch):
```

und mit:

```
das 2. Mal an (Kontroll-Eingabe):
```

nach den Überlade-Dateiangaben gefragt.

Zeilenabschlusszeichen werden bei der Eingabe von Überladeinformationen über Tastatur allgemein ignoriert bzw. in Form einer zusätzlichen Leerzeile als Ende der Überladeinformation interpretiert. Im Direktmodus werden jedoch auch Zeilenabschlusszeichen in Überladedateien als Teil der Überladeinformation aufgefasst.

Bat:/Dia: Bei mehreren Überladeinformationen empfiehlt sich fast immer eine entsprechende Überladedatei. Im aktuellen Programmlauf verwendete Überladedateien werden in diesem Programmlauf n i c h t behandelt (also auch nicht zerstört), selbst wenn diese explizit oder implizit als zu behandelnde Dateien angegeben wurden (siehe dazu 1.10. Ausnahmedatei oben).
Dia: In der „erweiterten" Version sind Überladedateien auch in einer der Metasyntaxformen möglich, in der „einfachen" Version wird eine Überladedatei als im Direktmodus vorliegend interpretiert.

Haben Sie im Checkbox-Feld „Überladedatei" ausgewählt, so erscheint das „Aktenmappe"-Symbol links neben dem Überladeinformations-Eingabefeld. Analog zur Beschreibung unter 1.6. Kontrollausgabedatei oben können Sie einen Dateinamen (mit Pfadangabe) in das Eingabefeld einstellen. Auch hier wird Ihnen in der Statusleiste angezeigt, wo sich Ihr Cursor befindet. Solange das Eingabefeld nicht ausgewählt ist, d.h. nicht den Fokus besitzt, werden die Dateinamen-Zeichen durch Sterne angezeigt. Besitzt das Eingabefeld den Fokus, ist der Feldinhalt nur dann sichtbar, wenn sich der Maus-Zeiger im Kernbereich des Eingabefeldes befindet bzw. bewegt. Wird der Maus-Zeiger langsam aus dem Kernbereich heraus bewegt, verschwindet der Feldinhalt wieder. Unabhängig davon sind markierte Feldteile immer sichtbar.

2.8 Überladedateien im Direktmodus

Bat:/Dia: Überladedateien im Direktmodus nehmen eine Sonderstellung bei der Schlüssel-Überladung ein.

Während es bei „kleinen" Schlüsseln im Zusammenhang mit CODING1 noch sinnvoll erscheint, die zu überladenden Stellen und deren Überladewerte einzeln (symbolisch) anzugeben, ist dieses Vorgehen bei den Schlüsselgrößen der beiden anderen Programme völlig ungeeignet, um einen „zufälligen" Schlüssel aus dem jeweils verfügbaren Schlüsselraum zu generieren.

Daher wurde speziell in Hinblick auf CODING2 und CODING3 das Verfahren der Schlüssel-Überladung erweitert.

Es ist möglich, mehrere Überladedateien (s. 1.9, 2.5) auszuwählen, die als eine Datenmenge betrachtet werden (s.1.11) **Dia:** (nur „erweiterte“ Version). Dazu kann im Überladeinformations-Eingabefeld für eine Dateinamen-Datei diese in runde Klammern (’(’, ’)’) gesetzt werden, ein Verzeichnisname steht in eckigen (’[’, ’]’) bzw. geschweiften (’{’, ’}’) Klammern. Mit Hilfe des „Aktenmappe“-Symbols kann dabei auch der Verzeichnis-Auswahl-Dialog gestartet und ein Verzeichnisname ausgewählt werden, falls das erste Zeichen im Eingabefeld eines der Klammern-Zeichen ’[’ oder ’{’ ist.

Des weiteren wird der Ausgangsschlüssel nicht nur einmal mit der Überlade-Datenmenge überladen sondern (ggf.) mehrfach, was im folgenden als **Iteration** der Datenmenge bezeichnet wird.

Ohne jeweilige Modifikation der Überlade-Datenmenge allerdings liefert auch dieser Ansatz keine all zu große „Zufälligkeit“, da zumindest die betroffenen Stellen im Ausgangsschlüssel, die durch die Datenmenge angesprochen werden, bei jedem Durchgang die gleichen bleiben würden.

Tatsächlich wird aber nur am Anfang die Überlade-Datenmenge selbst zur Schlüssel-Modifikation verwendet. Beim zweiten und folgenden Durchläufen wird die Überlade-Datenmenge vor dem jeweiligen Überlade-Vorgang **fortlaufend verschlüsselt** (!), wozu Schlüsselparameter verwendet werden, die ihrerseits aus dem Ausgangsschlüssel initiiert werden.

Diese Vorgehensweise hat zwei große Vorteile:

- § Unabhängig von der „Zufälligkeit“ und damit der Güte der Überlade-Datenmenge werden durch Anwendung des hier vorgestellten Chiffrierverfahrens auf die Überlade-Datenmenge hochgradig zufällige Überladeinformationen erzeugt.
- § Die Qualität der Überladung durch eine Datenmenge ist unabhängig von der Größe dieser Überlade-Datenmenge.

Wie Untersuchungen des Autors zeigen, kann nach ca. (x = Überladedatenmengengröße in Bytes)

1.410 / x (CODING1) **1.441.452 / x** (CODING2) **854.889.360 / x** (CODING3) **Iterationen** der abgeleitete Schlüssel nicht mehr von einem „zufälligen“ Schlüssel unterschieden werden.

Daher ist dem Programm für Überladedateien im Direktmodus noch der Iterationswert anzugeben:
Bat:

```
Geben Sie bitte die Anzahl der Iterationen an, die die Überladeinformationen
 durch den Schlüssel modifiziert werden sollen, wobei jeweils pro Iteration
 die geänderten Überladeinformationen auf den Schlüssel angewandt werden
  ('keine Iterationen'=0 oder 1 bis 99999999 (-1=Abbruch)):
```

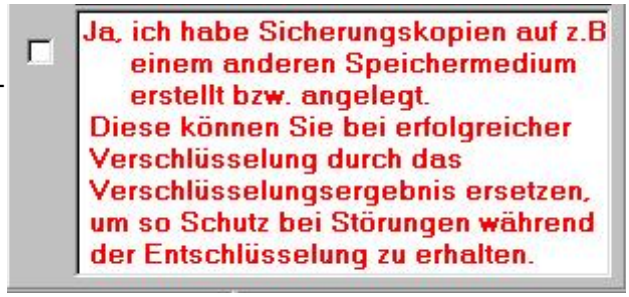
Dia: Der Feldinhalt im Iterationen-Feld verhält sich analog dem oben beschriebenen Überladeinformationsfeld im Fall von Überladedateien (siehe dort). Das Iterationen-Feld ist nur im Falle der Überladedatei-Behandlung im Direktmodus ansprechbar bzw. sichtbar.

Bat:/Dia: Speziell bei Überladedateien, die entweder relativ homogen sind oder die weniger als die oben genannten Zahlen Bytes umfassen, sollte der Iterations-Wert größer Null angegeben werden. Falls keine Stellen-, Überladewerte- und Schlüsselwerte-Intervall-Einschränkungen vorgenommen werden, sollte, um ca. 63,2% aller möglichen Schlüsselwerte auch tatsächlich im Schlüssel wiederzufinden (das ist etwa der Wert, der bei einer zufälligen Verteilung der Schlüsselwerte auftreten würde), bei einem nur „wenige“ Zeichen umfassenden Ausgangs-Schlüssel der Iterations-Wert wie oben beschrieben angegeben werden, wobei der Iterations-Wert bei relativ homogenen Überladedateien um 1 größer gewählt werden sollte. Falls Stellen-, Überladewert- und Schlüsselwert-Intervall-Einschränkungen existieren, so kann der so ermittelte Iterations-Wert ebenfalls als sinnvolle Grenze für „zufällige“ Variabilität angesehen werden.

Es steht Ihnen natürlich frei, einen größeren Iterations-Wert zu wählen (die genannten 63,2% werden davon nicht tangiert). Es sollte jedoch beachtet werden, dass bei großen Iterations-Werten die Schlüssel-Modifikation eine gewisse Zeit in Anspruch nimmt (die Überladedateien werden dabei (n+1)-mal gelesen und intern n-mal chiffriert).

3 Eigentliche Programm-Bearbeitung

Bat:/Dia: Da alle zu chiffrierenden Dateien direkt bearbeitet werden (es werden keinerlei Kopien erstellt), kann eine Störung des Programms in der Chiffrierphase ggf. eine oder auch mehrere Dateien **unwiederbringlich zerstören (!)**. Daher sollten Sie vorher **unbedingt Sicherungskopien** auf z.B. anderen Speichermedien erstellen bzw. anlegen (!). Diese können Sie bei erfolgreicher Verschlüsselung nochmals durch das Verschlüsselungsergebnis ersetzen, um so Schutz bei Störungen während der Entschlüsselung zu erhalten.



Dia: Um ein diesbezügliches unabsichtliches Versehen auszuschließen, **startet das Programm nur dann, wenn Sie die Erstellung von Sicherungskopien explizit bestätigt haben !**

Falls Sie Ihrer Meinung nach alle relevanten Informationen eingetragen und ausgewählt haben, können Sie den eigentlichen Programmlauf mit dem „Starten“-Button (siehe 1.5. Allgemeines) starten. Das Programm wird Sie ggf. zuvor nochmals auf unbestätigte Schlüssel- bzw. Überladeinformationen aufmerksam machen und weitere Warnungen ausgeben, so dass Sie den Bearbeitungsstart des Programms nochmals aufschieben können. Das Programm beendet den Startversuch auf jeden Fall dann mit einer genauen Fehlermeldung, wenn wichtige Informationen fehlen. Sie können nach Korrektur dieser Fehler einen erneuten Anstartversuch unternehmen.

Es werden jedoch in dieser Phase nicht alle Konsistenzen geprüft. Überlade- und/oder Parameterdateien werden erst zum Bearbeitungszeitpunkt der jeweiligen Datei einer genauen Prüfung unterzogen.

3.1 Programm-Initialisierung

Dia: Liegen von Seiten des Dialogsystems keine Beanstandungen vor, beginnt die eigentliche Arbeit des Programms, was nach kurzzeitiger Anzeige der Identifikationsseite (siehe 1.3. oben) im Protokollfenster durch die Startmeldung unter dem Programmphasen-Hinweis „Programm-Initialisierung ...“ angezeigt wird.

Wurde das Checkbox-Feld unter dem Eingabefeld für die Kontrollausgabedatei aktiviert (siehe 1.6. oben), wird lediglich der Hinweis „-- Stille ! --“ in das Protokollfenster eingestellt und mit Abschluss der erfolgreichen Bearbeitung wird ein einzelner Punkt in die Folgezeile geschrieben. Weitere Protokollfenster-Meldungen unterbleiben in diesem Fall, soweit keine Fehler auftreten (!).

Bat:/Dia: Falls bei eingeschalteter Protokollierung (siehe 1.6. oben) eine Verarbeitung von Überladeinformationen vorzunehmen ist, so wird dies mit folgendem Hinweis angezeigt:

```
Beginn der Überlade-Informations-Behandlung: Do. 07.12.2006, 14:28:13.
```

bei Fortsetzung dieser Verarbeitung erscheint:

```
Fortsetzung der Überlade-Informations-Behandlung: Do. 07.12.2006, 14:28:13.
```

und das Ende dieses Abschnitts wird wie folgt angezeigt:

```
Ende der Überlade-Informations-Behandlung: Do. 07.12.2006, 14:28:13.
```

Der Initialisierungs-Abschnitt des Programms wird bei eingeschalteter Protokollierung mit folgendem Hinweis eingeleitet:

```
Beginn der Initialisierung: Do. 07.12.2006, 14:28:13.
```


Die Ausführlichkeit, mit der die CODING-Programme die Aktionen innerhalb dieses und ggf. des Überladeinformations-Abschnitts kommentieren, ist unterschiedlich. Da bei CODING3 eine Tabelle, ein Schlüssel, usw. 16.777.216 Elemente á 3 Bytes umfasst, die analogen Informationen in CODING1 aber nur 256 Bytes, werden in CODING3 die Überlade- und Initialisierungsarbeiten sehr viel ausführlicher angezeigt als in CODING1.

Die Programme CODING1 und CODING2 sind daher auch jeweils nur nach einem Iterationsschritt bei der Überladeinformations-Bearbeitung durch Zeitgrenzenüberschreitung unterbrechbar, während CODING3 zusätzlich auch während der Initialisierung von Chiffrierparametern die jeweilige Zeitgrenze überprüft und damit unterbrochen werden kann.

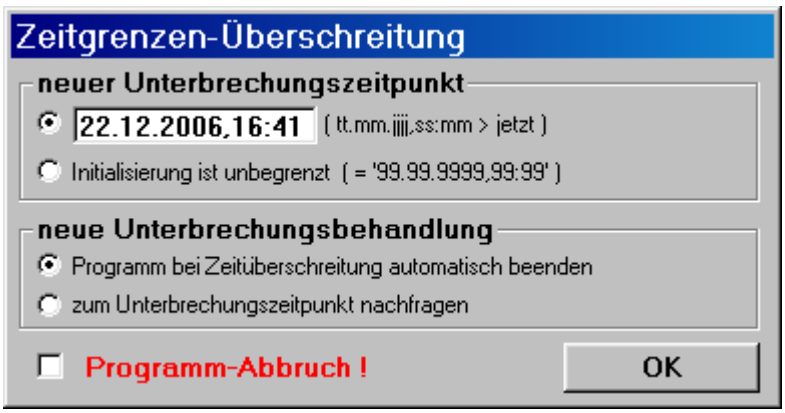
Den Programmen gemeinsam sind Informationen und Nachfragen für den Fall, dass eine Parameterdatei-Bearbeitung vorliegt.

Falls bei der Initialisierung einer Parameterdatei die angegebene Zeitgrenze überschritten und die Nachfrage des Programms gewünscht wurde, so fragt das Programm wie folgt an:

Bat:

```
Soll die Bearbeitung mit der Initialisierung fortgesetzt werden ?
   0    - Nein, das Programm soll beendet werden,
   1    - Ja, das Programm soll mit der Initialisierung fortsetzen.
Geben Sie bitte die zugehörige Zahl an:
```

Danach erfolgt eine erneute Abfrage der Zeitgrenze (siehe 1.8.4. oben) sowie der Aktion bei Zeitgrenzen-Überscheitung (siehe 1.8.5. oben), falls die Fortsetzung des Programms bestätigt wurde.



Dia: Neben den schon bekannten Parametern der Zeitgrenze (siehe 1.8.4. oben) sowie der Aktion bei Zeitgrenzen-Überscheitung (siehe 1.8.5. oben) beinhaltet die Dialogmaske noch ein Checkbox-Feld für den vorzeitigen Programmabbruch. Mit dem „OK“-Button setzen Sie die den Programmlauf fort, entweder in dem die laufende Bearbeitung vorzeitig beendet wird oder in dem diese Bearbeitung mit den von Ihnen neu angegebenen Parametern fortgesetzt wird.

Bat:/Dia: Vor der Erzeugung und abschließenden Speicherung der Parameter zur Chiffrierung von Daten in eine Parameterdatei wird die Güte des Ausgangs- oder des abgeleiteten Schlüssels, aus dem diese Parameter erzeugt werden, in Form einer Statistik ausgegeben (hier: CODING1):

```
  Statistik-  +--------- Zeichen-Häufigkeiten in %: ---------+ % binäre  Null-
   Inhalt:    min.:        max.:      Mittelw.:     Std.Abw.:    1en:  Gruppen:
-------------------------------------------------------------------------------
  Schlüssel:  0,00000000  1,56250000 (0,39062500)   0,38832041 49,36523437  93
```

Dabei gilt, dass ein Schlüssel um so weitgehender einem „Zufallsschlüssel“ entspricht,

- je näher die Anzahl der Null-Gruppen, d.h. die Anzahl der Zeichen (Elemente), die nicht im Schlüssel auftreten, am Wert 94 bzw. 24.101 bzw. 6.172.493 (CODING1/2/3) aller möglichen Zeichen (Elemente) liegt,
- je dichter der Prozentwert binärer Einsen an 50% liegt,
- je dichter die maximale Zeichen-Häufigkeit am Mittelwert liegt,
- je kleiner die Standardabweichung ist.

Bat: Falls der Initialisierungsprozess einer Parameterdatei abgeschlossen und die Nachfrage des Programms gewünscht wurde, so wird für den Fall, dass eine Chiffrierung von Daten vorgenommen werden soll, folgende Nachfrage gestellt:

```
Soll die Bearbeitung mit der Kodierung fortgesetzt werden ?
   0    - Nein, das Programm soll beendet werden,
   1    - Ja, das Programm soll mit der Kodierung fortsetzen.
Geben Sie bitte die zugehörige Zahl an:
```

Bat:/Dia: Das Ende der Initialisierung wird bei eingeschalteter Protokollierung wie folgt ausgegeben:

```
Ende der Initialisierung: Do. 07.12.2006, 14:28:13.
```

3.2 Programm-Chiffrierung

Nach Abschluss der Initialisierung, – auch eine vollständig initialisierte Parameterdatei benötigt eine die Transformationstabelle(n) aktivierende Initialisierung (siehe 1.13. oben) –, und falls keine reine Parameterdatei-Bearbeitung vorliegt, beginnt das Programm mit der eigentlichen Chiffrierung der Daten (z.B.):

```
Beginn der Ver-/Entschlüsselung: Do. 07.12.2006, 14:28:13.
```

Damit ist das Programm **n i c h t** mehr unterbrechbar. Da alle Dateien direkt bearbeitet werden (es werden keinerlei Kopien erstellt), würde eine Unterbrechung ggf. eine oder auch mehrere Dateien **unwiederbringlich zerstören (!)**, d.h. Sie sollten vorher **unbedingt Sicherungskopien** auf z.B. anderen Speichermedien erstellt bzw. angelegt haben (!). Diese können Sie bei erfolgreicher Verschlüsselung nochmals durch das Verschlüsselungsergebnis ersetzen, um so Schutz bei Störungen während der Entschlüsselung zu erhalten.

Falls Sie die Protokollierung nicht ausgeschaltet haben (siehe 1.6. oben), werden bei der Bearbeitung alle behandelten Dateien und Verzeichnisse jeweils auf dem Bildschirm / Protokollfenster und in der oben angegebenen Protokolldatei mitprotokolliert, so dass jeder Schritt aktuell und auch später noch von Ihnen verfolgt werden kann.

Wurde von Ihnen eine Daten-Gesamtstatistik angefordert (siehe 1.12. oben), so wird diese am Ende der Bearbeitung der Daten ausgegeben (hier: CODING1):

```
  Statistik-  +--------- Zeichen-Häufigkeiten in %: ---------+ % binäre  Null-
   Inhalt:    min.:        max.:      Mittelw.:     Std.Abw.:    1en:  Gruppen:
-------------------------------------------------------------------------------
alle Dateien: 0,38749079  0,39417626 (0,39062500)   0,00130741 49,99381021   0
```

Mit Angaben zu behandelten Zeichen, Dateien und Verzeichnissen sowie der Bearbeitungs-Ende-Meldung wird der Daten-Bearbeitungs-Abschnitt des Programms beendet (z.B.):

```
    212.430.600 Bytes in  54 Datei(en) aus  9 Verzeichnis(sen) bearbeitet.
Ende der Bearbeitung: Do. 07.12.2006, 14:28:35.
```

Mit der Programm-Ende-Zeile

```
Ver-/Entschlüsselungsprogramm  C O D I N G 1  1.1  beendet.
```

verabschiedet sich der Bearbeitungsteil des Programms.

Dia: Mit Abschluss einer Bearbeitung werden die Operations-Button (siehe 1.5. oben) wieder eingeblendet. Mit dem „Ende“-Button können Sie jetzt die Anwendung wieder verlassen, mit dem„Rücksetzen“-Button können alle Felder einschließlich des Protokollfensters wieder geleert bzw. auf ihre Standardwerte zurückgesetzt werden. Falls dies nicht geschieht, bleiben die

Meldungen des abgeschlossenen Bearbeitungslaufs auch bei einem weiteren Bearbeitungslauf erhalten, sonst identische Dateien werden jedoch überschrieben bzw. ggf. erneut modifiziert (!).

Falls Sie den Maus-Zeiger in den Bereich des **Protokollfenster**s bewegen und dieses mehr als 2 Meldungszeilen aufweist, wird dieses nach unten rechts bis zu den Grenzen des Haupteingabe-formulars **aufgeblendet**, so dass sich die Meldungen wesentlich besser lesen lassen. So bald der Maus-Zeiger das Protokollfenster wieder verlässt, schrumpft das Protokollfenster wieder auf seine ursprüngliche Größe.

4 Erweiterungen des Standard-Programms

4.1 Stromchiffrierung

Bat:/Dia: Das bisher beschriebene Programm kann in der „erweiterten“ Version durch die Möglichkeit der **Stromchiffrierung** speziell für das professionelle Umfeld nochmals wesentlich erweitert werden. Dabei handelt es sich um die Möglichkeit, einen erreichten Parameterzustand zu speichern, um bei der Chiffrierung weiterer Dateien wieder auf diesem Zustand aufzusetzen und diesen fortzuführen. Ein Parametersatz kann dabei potentiell **sowohl zur Ver- als auch zur Entschlüsselung** verwendet werden.

Dia:

Bat: Nach der Bearbeitungsform (siehe 1.11) wird hierbei ergänzend folgende Frage gestellt:

```
Soll die Bearbeitung in Stromchiffrierungsform durchgeführt werden?
   0  nein, es wird  k e i n e  Stromchiffrierung durchgeführt,
   1  ja, es wird Stromchiffrierung durchgeführt.
Geben Sie bitte die zugehörige Zahl an:
```

oder:

```
Soll die Bearbeitung in Stromchiffrierungsform durchgeführt werden?
   0  nein, es wird  k e i n e  Stromchiffrierung durchgeführt,
   1  ja, es wird Stromchiffrierung unter Verwendung
      einer "normalen" Parameterdatei durchgeführt,
   2  ja, es wird Stromchiffrierung unter Verwendung
      einer Komponentenwechsel-Parameterdatei durchgeführt,
   3  ja, es wird Stromchiffrierung unter Verwendung
      sowohl einer "normalen" Parameterdatei
      als auch einer Komponentenwechsel-Parameterdatei durchgeführt.
Geben Sie bitte die zugehörige Zahl an:
```

Bat:/Dia: Eine „normale“ Parameterdatei, die zur Speicherung des Stromchiffrierungszustands verwendet wird, wird im Unterschied zur Aussage unter 1.8.1 Parameterdatei zwar nicht als normale Datei aber dennoch verändert, da in ihr der erreichte Parameterzustand am Ende neu gespeichert wird. Um daher auf einen gesicherten Stand zurückgreifen zu können, sollte vor Verarbeitung eine Kopie der Parameterdatei gespeichert werden, die nach erfolgreicher Verarbeitung wieder mit der neu erzeugten Parameterdatei überschrieben bzw. gelöscht werden kann.

Des weiteren ist zu berücksichtigen, dass sowohl für die Verschlüsselung als auch für die Entschlüsselung einer Datenmenge **getrennte und damit eigene Parameterdateien** notwendig sind.

Stromchiffrierung ist sicher dann anzuwenden, wenn

- zum einen eine **erhöhte Sicherheit** gewünscht wird,
- zum anderen ein **Datenaustausch** über eine unsichere (öffentliche) Verbindung zwischen zwei oder mehreren Beteiligten stattfinden soll (z.B. via **Internet**).

Dabei kann der Austausch von Schlüsseln auf ein Minimum beschränkt werden. Denn im Gegensatz zum Standard-Programm, für das die Chiffrier-Komponenten nur zu Anfang aus den Schlüssel-Informationen ermittelt werden und nachfolgend nur dann, wenn ein Pseudo-Zufallszahlen-Zyklus beendet wird (d.h. sich die Pseudo-Zufallszahlen wiederholen würden), lässt sich die Durchführung solcher Komponentenwechsel im entsprechend erweiterten Programm (verschärfte Chiffrierung) genauer steuern (siehe unten 5.2 Algorithmus-Parameter).

Allgemein wird bei einem Komponentenwechsel der aktuell verwendete interne Operationsschlüssel als Ausgangsschlüssel für den Komponentenwechsel verwendet. Aus den daraus resultierenden Chiffrier-Komponenten kann der dazu verwendete Ausgangsschlüssel nicht (rückwärts) hergeleitet werden (siehe dazu auch „Kurz-Untersuchung zur Sicherheit des Algorithmus“).

Sollte daher ein Komponentensatz kompromittiert werden, so bleiben zunächst alle Daten unkompromittiert, die vor dem letzten Komponentenwechsel chiffriert wurden, nicht jedoch alle weiteren Daten.

Um auch für die Daten nach dem nächsten Komponentenwechsel den Kompromittierungsschutz nicht zu verlieren (die mit dem kompromittierten Komponentensatz chiffrierten Daten sind natürlich auch kompromittiert), kann – falls das Programm auf der höchsten Chiffrierstufe arbeitet – der Ausgangsschlüssel eines Komponentenwechsels selbst vor Erzeugung seines Komponentensatzes mit Hilfe einer zweiten Parameterdatei, der **Komponentenwechsel-Parameterdatei**, „chiffriert“ werden.

Falls diese Barriere von unbefugter dritter Seite (ohne direkten Zugriff auf die Komponentenwechsel-Parameterdatei) durchbrochen werden soll, müsste dieser Angreifer zunächst „sehr viele“ Komponentensätze der „normalen“ Parameterdatei kompromittieren, bevor er nur ansatzweise eine Möglichkeit hätte, an den Komponentensatz dieser zweiten Parameterdatei heranzukommen.

Allgemein wird bei Ver- bzw. Entschlüsselung **einzelner Dateien** (siehe 1.11 Bearbeitungsform oben) und Stromchiffrierung ein Komponentenwechsel am Anfang **jeder** Datei vorgenommen.

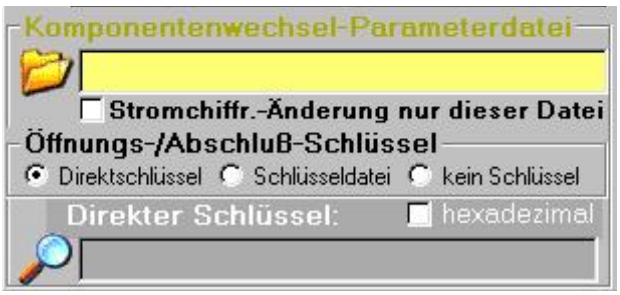
Sollen die zu behandelnden Dateien als eine **Datenmenge** gesehen werden und soll nur der Parameterzustand der Komponentenwechsel-Parameterdatei am Ende gespeichert werden, so wird am Anfang jeder Datenmengen-Bearbeitung ein Komponentenwechsel durchgeführt. Dies ist zwingend erforderlich, da nur dadurch adäquate Wiederaufsetzpunkte bei Bearbeitung mehrerer Datenmengen gegeben sind (der Zwischenstand der üblichen Parameter kann nicht gespeichert werden, da hierfür keine „normale“ Parameterdatei zur Verfügung steht !).

Sollen die zu behandelnden Dateien als eine **Datenmenge** gesehen werden und wird neben einer Komponentenwechsel-Parameterdatei auch eine „normale“ Parameterdatei zur Speicherung des Bearbeitungsstands der Stromchiffrierung verwendet, so wird am Anfang der ersten Datenmengen -Bearbeitung, d.h. falls die „normale“ Parameterdatei nach Initialisierung noch unmodifiziert ist (angezeigt durch den **Stromchiffrierungs-Behandlungswert** der Parameterdatei), zunächst ein Komponentenwechsel durchgeführt, um schon beim ersten verwendeten Komponentensatz die Abhängigkeit von der Komponentenwechsel-Parameterdatei sicher zu stellen (im Folgenden ist dies dann automatisch der Fall).

4.2 Komponentenwechsel-Parameterdatei

Bat:/Dia: Eine Komponentenwechsel-Parameterdatei ist zunächst eine „normale“ Parameterdatei mit dem Zusatz, dass sie im betrachteten Programmlauf nur zur Chiffrierung von Operationsschlüsseln eingesetzt wird.

Dia:



Bat: Daher fragt das Programm (in der höchsten Chiffrierstufe) zunächst nach der Verwendung einer Komponentenwechsel-Parameterdatei:

```
Soll eine Komponentenwechsel-Parameterdatei verwendet werden?
  '0'   - Nein, keine Komponentenwechsel-Parameterdatei verwenden,
  'C'   - Ja, verwende eine Komponentenwechsel-Parameterdatei.
  '9'   - Sofortiger Programmabbruch.
Geben Sie bitte die zugehörige Zahl bzw. Zeichen an:
```

Wurde diese Frage bejaht, erfolgt die Aufforderung zur Eingabe des Dateinamens:

```
Geben Sie bitte den Dateinamen der Komponentenwechsel-Parameterdatei an
  (Leereingabe = Programmabbruch):
```

Bat:/Dia: Eine im aktuellen Programmlauf verwendete Komponentenwechsel-Parameterdatei wird wie eine „normale“ Parameterdatei in diesem Programmlauf n i c h t als normale Datei behandelt, aber dennoch verändert, da in ihr der erreichte Parameterzustand gespeichert wird. Um daher auf einen gesicherten Stand zurückgreifen zu können, sollte vor Verarbeitung eine Kopie der

Komponentenwechsel-Parameterdatei gespeichert werden, die nach erfolgreicher Verarbeitung wieder mit der neu erzeugten Komponentenwechsel-Parameterdatei überschrieben bzw. gelöscht werden kann.

Wie eine „normale“ Parameterdatei kann auch eine Komponentenwechsel-Parameterdatei mit einem Schlüssel gesichert sein, der bei der Chiffrierung des abschließend vorhandenen Parameterzustands erneut Verwendung findet. Ist dies der Fall, werden Sie, wie unter 1.13 Schlüssel-Informationen speziell in den Abschnitten 1.13.1 bis 1.13.4 beschrieben, zur Eingabe der entsprechenden Parameter aufgefordert bzw. können Sie die Angaben in die zuvor angegebenen Maskenteile eintragen.

Dia: Um bei Stromchiffrierung auch eine „normale“ Parameterdatei unverändert lassen zu können (keine Speicherung des Stromchiffrierungszustands in diese), kann das entsprechende Checkbox-Feld direkt unter dem Komponentenwechsel-Parameterdateinamen aktiviert werden.

4.3 Stromchiffrierungs-Behandlungswert

Bat:/Dia: Auch der zuvor schon erwähnte Stromchiffrierungs-Behandlungswert kann explizit verändert – invertiert – werden.

Bat: In 1.13.1 Abfrage zur Parameterdatei-Sicherung wird daher – falls das Programm auf der höchsten Chiffrierungsstufe arbeitet – wie folgt nachgefragt:

```
Soll die Parameterdatei mit einem Sicherungsschlüssel gesichert werden?
  '0'   - Nein, keine Parameterdatei-Sicherung,
  'S'   - Ja, Sicherung der Parameterdatei mit einem eigenen Schlüssel.
  'R'   - Ja, Sicherung der Parameterdatei mit einem eigenen Schlüssel
          und Invertierung ihres Stromchiffrierungs-Behandlungswertes
  '9'   - Sofortiger Programmabbruch.
Geben Sie bitte die zugehörige Zahl bzw. Zeichen an:
```

Bat:/Dia: Die Änderung des Stromchiffrierungs-Behandlungswertes ist nur dann möglich, wenn bei der Parameterdatei ein Wechsel ihres Sicherungsschlüssels durchgeführt wird.

4.4 Kompromittierungs-Schlüssel

Bat:/Dia: Wie in 1.13 Schlüssel-Informationen bereits beschrieben, kann bei Entschlüsselungs-versuchen von gesicherten Parameterdateien nicht auf Grund des dabei entstehenden Parameterdatei-Ergebnisses entschieden werden, ob der Entschlüsselungsversuch erfolgreich war. Dies wird intern dadurch sicher gestellt, dass alle in irgend einer Form „verräterischen“ Informationen vor dem Speichern in eine Parameterdatei in eine Form gebracht werden, die **jeden** darin enthaltenen Wert als gültigen Wert regulär vorkommen lassen.

Selbst wenn daher der im Programm realisierte Algorithmus diesbezüglich „überwacht“ beobachtet wird („trace“), kann **unter keinen Umständen** entschieden werden, ob ein „korrekter“ oder „falscher“ Sicherungsschlüssel für eine Parameterdatei eingegeben wurde !

Dies eröffnet die Möglichkeit, bei der Stromchiffrierung verwendete Parameterdateien auf zwei (oder auch mehr) unterschiedliche Arten zu verwenden. Ist eine beteiligte Stelle nicht mehr „Herr ihrer selbst“, so kann sie dies den sonstigen Beteiligten dadurch mitteilen, dass sie eine Parameterdatei nicht mehr in der gewöhnlichen Art und Weise verwendet, sondern dass sie mit Hilfe eines zuvor vereinbarten **Kompromittierungs-Schlüssels** auf einen Parametersatz übergeht, der ihren „kompromittierten“ Zustand anzeigt. Im Online-Fall könnte dieser Übergang sendeseitig z.B. auch durch das Betätigen einer Tasten-Kombination veranlasst werden.

Der oder die Empfänger stellen fest, dass der normale Parametersatz des Senders versagt und können durch Verschlüsselung mit dem üblichen Sicherungsschlüssel und Entschlüsselung mit dem Kompromittierungs-Sicherungsschlüssel auf den Kompromittierungs-Parametersatz des Senders übergehen und es damit versuchen. Gelingt dies, werden die Empfänger den Sender als kompromittiert wahrnehmen und ihre Inhalte entsprechend anpassen.

4.5 Störungen des Datenaustauschs

Für den oder die Empfänger kann sich eine Situation ergeben, in der die Daten zunächst ab einem bestimmten Datenblock nach der Entschlüsselung keine sinnvollen Ergebnisse mehr liefern. Da die Parameter-Entwicklung bei den beteiligten Stellen absolut parallel verläuft, gibt es neben der Kompromittierung des Senders hierfür nur die Möglichkeit

- § eines Übermittlungsfehlers, d.h. ein oder mehrere Datenblöcke sind beschädigt,
- § es wurde ein falscher Sicherungsschlüssel verwendet,
- § es wurden von dritter Seite Datenblöcke in den Datenstrom eingeschleust oder entfernt oder beschädigt, um die Kommunikation zu torpedieren.

Da der Chiffrierverfahrens-Fortschritt **nicht** an Blockinhalte gekoppelt ist, ist im ersten und dritten Fall eine **Resynchronisation potentiell möglich**, indem bei Übermittlungsfehlern einfach normal weiter gearbeitet wird oder sonst Datenblöcke übersprungen oder die Chiffrierparameter virtuell auf nicht vorhandene Datenblöcke angewendet werden.

Da das hier vorliegende Programm zur Zeit jedoch nur für den Offline-Betrieb (keine direkte Kommunikation) ausgelegt ist, bleibt die entsprechende Umsetzung einem Online-Programm vorbehalten.

4.6 Weitere potentielle Anwendungsmöglichkeiten

Der hier vorliegende Algorithmus chiffriert die Datenblöcke unabhängig voneinander. Im hier vorliegenden Verfahren kann eine (fast) beliebige oder sogar variable Blocklänge gewählt werden (siehe 5.2 Algorithmus-Parameter).

Das Verketten von Datenblöcken analog dem CBC-Modus (Cipher Block Chaining) des DES-Verfahrens erachtet der Autor nicht als sehr sinnvoll, da damit nicht nur beim Datenaustausch durch einen einzigen Fehler der Abbruch eines Datenstroms und ein Wiederaufsetzen erzwungen wird mit allen hiermit verbundenen Problemen und Kompromittierungsmöglichkeiten. Die Sicherheit des vorliegenden Verfahrens lässt die Blockverkettung als unnötig erscheinen, zur Übermittlungs-Sicherung sollten Fehler erkennende und korrigierende Kodierungen (z.B. Huffman-Code) eingesetzt werden.

Der CFB-Modus (Cipher Feedback) des DES-Verfahrens dient z.B. zur Erzeugung von **Pseudozufallszahlen**. Wie den Ausführungen in „Konvergenz-Eigenschaft des zentralen Operators XOR“ zu entnehmen ist, wird mit dem vorliegenden Verfahren nicht auf die Erzeugung aller möglichen Zustände eines Blockes abgezielt, sondern auf solche Blockzustände, in denen die Verteilung von Bits, Bytes bzw. Elementen (Byte-Gruppen) möglichst „gleichmäßig“ ist. In wie fern diese Blockinhalte durch geeignete Modifikationen zu „brauchbaren“ Pseudozufallszahlen gemacht werden können, bleibt zukünftigen Untersuchungen vorbehalten.

Der OFB-Modus (Output Feedback) des DES-Verfahrens arbeitet unabhängig von einem Blockinhalt und wird u.a. zur **Authentisierung** eingesetzt. Hierfür kann der vorliegende Algorithmus auch für (fast) beliebige Länge der Authentisierung (entspricht der gewählten Blocklänge) eingesetzt werden, falls man einfach einen wahlfreien aber festen Blockinhalt vorgibt. Nach den Untersuchungen in „Kurz-Untersuchung zur Sicherheit des Algorithmus“ besteht keinerlei Möglichkeit, Rückschlüsse auf die Chiffrierparameter oder gar den Ausgangsschlüssel zu ziehen. Auf Basis des vorliegenden Algorithmus sind daher „sehr viele“ verschiedene Authentisierungsverfahren möglich.

Darüber hinaus und in Analogie zu dem gerade gesagten lassen sind mit dem vorliegenden Verfahren „beliebige“ **Hash-Verfahren** mit (fast) beliebigen Hash-Resultatlängen implementieren. Denn wie auch bei der Authentifizierung kommt es bei Hash-Resultaten nicht darauf an, dass – wie bei Pseudozufallszahlen – alle Resultatzustände auch tatsächlich auftreten. Dabei wird der bisherige Inhalt des (einzigen) Blocks einfach vor erneuter Anwendung des Operationsschlüssels mit diesem „verknüpft“ (z.B. per ’xor’), bevor der nächste Datenblock behandelt wird. Das Ergebnis des letzten so behandelten Blocks, der ggf. zuvor auf die Hash-Länge (gleich Blocklänge) – wie auch immer – erweitert wurde, ist der Hash-Code der Daten.

5 System- und Algorithmus-Parameter des Programms

Die beschriebenen Programme und Komponenten können, durch interne Parameter gesteuert, auf die verschiedensten Anforderungen angepasst werden.

5.1 System-Parameter

Hierunter fallen alle Parameter, die Anpassungen auf Rechner, Betriebssystem und -art betreffen.

5.1.1 Gruppe

=1,2,3 - Anzahl der Bytes pro Bytegruppe (Element) (liegt für CODING1/2/3 fest)

5.1.2 Modulart

=0 - Hauptprogramm
=1 - Unterprogramm für GUI
=2 - Unterprogramm in DLL

5.1.3 DateiSystem

=0 - Windows (auch >4 GB)
=1 - Unix ohne Logik für Dateien >2GB
=2 - Unix mit Logik für Dateien >2GB
- - - - Host-Dateisystemlogik noch nicht integriert

5.1.4 ZeichenSatz

=0 - ASCII
=1 - EBCDIC (z.B. Host-Computer)

Bei Übertragungen verschlüsselter Dateien von ASCII auf EBCDIC oder umgekehrt dürfen keinerlei Zeichen-Umsetzungen stattfinden ! Im Zweifelsfall können die Dateien vor Übertragung expandiert und nach Übertragung deexpandiert werden. Nach der Entschlüsselung auf dem Zielsystem muss für Dateien mit „lesbarem“ Inhalt noch nachträglich eine explizite Transformation auf den Ziel-Rechner-Zeichensatz vorgenommen werden.

5.1.5 BSigRtL

Double/Integer-Werte-Signifikanz der Mantissen-/Integer-Bytes zunehmend (der Exponent folgend)

=0,2 - von links nach rechts (z.B. PC-Prozessor)
=1,3 - von rechts nach links (z.B. Großrechner-CPU)

Ausgabe (und Eingabe) des Programms ist auf jeden Fall die Byte-Reihenfolge, die für

=0,1 - bei =0
=2,3 - bei =3 entsteht („korrekte Byte-Reihenfolge“) bzw. entstanden wäre

CODING2/3: Da fast alle Operationen Element- und nicht Byte-weise durchgeführt werden, werden alle Eingangsdaten wie Schlüssel und Dateidaten für =1/=2 zunächst von „korrekte Byte-Reihenfolge“ auf „korrekte Element-Wertigkeit“ umgeschaltet, um erst vor der Ausgabe wieder auf „korrekte Byte-Reihenfolge“ umgesetzt zu werden. Ausnahmen hiervon bilden Byte-Operationen im Zusammenhang mit Element-Resten, die nur im Zustand „korrekte Byte-Reihenfolge“ ausgeführt werden können und die Parameterdatei-Behandlung.

Falls (bei gleichen Algorithmus-Parametern, sonstige System-Parameter beliebig) alle Rechner mit =0/=1 o d e r mit =2/=3 arbeiten (andere Kombinationen sind n i c h t möglich), dann können Dateien auf einem Rechner der einen Art ver- und auf einem Rechner der anderen Art entschlüsselt werden. Unter derselben Werte-Kombination können auch Parameterdateien auf Rechnern der einen oder auch anderen Art erstellt, vervollständigt und/oder verwendet werden.

5.1.6 Verpflichtung

=0 - ohne Verpflichtung
=1 - mit Verpflichtung und Zustimmung

5.2 Algorithmus-Parameter

Mit Hilfe der nachfolgend aufgeführten Parameter kann der Algorithmus an eine große Bandbreite von Anforderungen sowohl von technischer als auch von sicherheitsrelevanter Seite angepasst werden.

5.2.1 **varBlk** (CODING1/2)

=0 - feste Block-Länge
=1 - variable Block-Länge mit max. Block-Länge = 2 * min. Block-Länge

5.2.2 **BlockElem**

=n - feste/minimale (Daten-)Block-Länge in Elementen (Bytegruppen);
- m u s s >1 und <2**(16*Gruppe) sein und
- m u s s >=8/4/4 (CODING1/2/3) sein, falls variable Block-Länge

=512,256,16384 - aktuelle Werte für CODING1/2/3

5.2.3 **BlkKey**

Für feste Block-Länge (CODING1/2: 'varBlk'=0):
=0 - Operations-Schlüssel-Länge = Block-Länge
=1 - Block-Länge >= Operations-Schlüssel-Länge
=2 - Operations-Schlüssel-Länge >= 2 * Block-Länge
=3 - Block-Länge < Operations-Schlüssel-Länge < 2 * Block-Länge

CODING1/2: Für variable Block-Länge ('varBlk'=1):
=1 - min. Block-Länge >= Operations-Schlüssel-Länge
=2 - Operations-Schlüssel-Länge >= 2 * max. Block-Länge
=3 - min. Block-Länge < Operations-Schlüssel-Länge < 2 * max. Block-Länge

CODING3:
=4 - 2 + 'BlkKey2'=2 + variable Block-Länge
mit max. Block-Länge = 2 * min. Block-Länge
(größere max. Block-Längen sind z.Zt. nicht sehr sinnvoll und daher nicht realisiert)

5.2.4 **BlkKey2** (CODING3)

=0 - keine Relevanz (’BlkKey’<>2 bzw. <>4)
=1 - Operations-Schlüssel-Länge in Elementen < 3 * Block-Länge in Elementen
=2 - Operations-Schlüssel-Länge in Elementen >= 3 * Block-Länge in Elementen

5.2.5 **Safety**

=0 - **Standard-Chiffrierung**:
- keine Linearoperations-Wiederholung nebst Verschiebung bei Daten-Blöcken und Parameterdatei,
- statische Linearoperations-Wiederholung bei der Operations-Schlüssel-Modifikation bei Komponentenwechseln,
- Komponentenwechsel nur bei vollständigem Zufallszahlenzyklus-Durchlauf,
- statische lineare Modifikation des Operations-Schlüssels nach einer durchschnitt lichen Anzahl von 'MeanBlksD' Blöcken für Daten und 'MeanBlksO' Blöcken für Überladedaten (Operations-Schlüssel-Modifikation vor j e d e m Block, falls 'MeanBlks_'=0),
- keine Verwendung einer Komponentenwechsel-Parameterdatei,
- keine Stromchiffrierung.

=1 - **verschärfte Chiffrierung**:
- variable Linearoperations-Wiederholung nebst Verschiebung bei Daten-Blöcken und Parameterdatei,

- variable Linearoperations-Wiederholung bei der Operations-Schlüssel-Modifikation bei Komponentenwechseln,
- Komponentenwechsel nach einer durchschnittlichen Anzahl von 'MeanBlksD' Blöcken für Daten und 'MeanBlksO' Blöcken für Überladedaten (Komponenten wechsel vor j e d e m Block, falls 'MeanBlks_'=0),
- statische lineare Modifikation des Operations-Schlüssels nach Abarbeitung des Operations-Schlüssels für jeden sonstigen Block,
- keine Verwendung einer Komponentenwechsel-Parameterdatei,
- Möglichkeit der Stromchiffrierung bei Verwendung einer Parameterdatei mit Speicherung des letzten Chiffrierungszustands in der Parameterdatei.

=2 - **höchste Chiffrierungsstufe**:
- verschärfte Chiffrierung (siehe =1),
- Komponentenwechsel ggf. unter Verwendung einer zusätzlichen Komponenten wechsel-Parameterdatei mit Speicherung des letzten Komponentenwechsel-Zustands in der Komponentenwechsel-Parameterdatei,
- Komponentenwechsel dieser Komponentenwechsel-Parameterdatei nach einer durchschnittlichen Anzahl von 'MeanBlksX' Operationsschlüssel-Blöcken (Komponentenwechsel dieser Komponenten vor j e d e r Operationsschlüssel-Modifikation, falls 'MeanBlksX'=0),
- statische lineare Modifikation des Operations-Schlüssels der Komponenten wechsel-Parameterdatei vor jeder sonstigen Operationsschlüssel-Modifikation.

=3 - **höchste Chiffrierungsstufe**:
- auch die statische lineare Modifikation des Operations-Schlüssels wird als "Komponentenwechsel" aufgefasst.

5.2.6 stLoop

=n - statischer Linearoperations-Wiederholungswert >0 CODING3: und < 1000
=2 - aktueller Wert

5.2.7 stLoopC

=n - statischer Linearoperations-Wiederholungswert >0 bei der Modifikation des Operations-Schlüssels der Komponentenwechsel-Parameterdatei
=2 - aktueller Wert

5.2.8 addLoop

=n - Erhöhungswert des Linearoperations-Wiederholungswertes für Datenblöcke; dieser Wert erhöht den entsprechenden Wiederholungswert 'Loop', der sich aus der Blocklänge in Elementen wie folgt ergibt:
Loop := $(\log_2(\text{Blockelemente})+3)/4$ + Wert aus $[0,(\log_2(\text{Blockelemente})+3)/4]$
=0 - aktueller Wert

5.2.9 opLoop

=n - Basis- und max. Erhöhungswert für Linearoperations-Wiederholungen bei der Modifikation von Operations-Schlüsseln in Zusammenhang mit Komponentenwechsel,
- >= $\log_2$(Schlüsselelemente)/4 ist sinnvoll, CODING3: < 500
=2,4,6 - aktuelle Werte für CODING1/2/3

5.2.10 PDLoop

=n - Basis- und max. Erhöhungswert für Linearoperations-Wiederholungen bei der Chiffrierung eines Parameterblocks
- > $\log_2$(2*Schlüsselelemente)/4 ist sinnvoll
=3,5,7 - aktuelle Werte für CODING1/2/3

5.2.11 EndShift

=0 - keine Datenblock-Endverschiebung
=1 - mit Datenblock-Endverschiebung

5.2.12 ClearCore

=0 - keine explizite Löschung von verwendeten Hauptspeicherbereichen vor Programmende
=1 - verwendete Hauptspeicherbereiche werden vor Programmende explizit gelöscht

Jeder der folgenden 3 Werte identifiziert die zu berücksichtigenden Wiederholungswerte-Bits, d.h. identifiziert die Bitstellen, die im 2. Zufallszahlen-Doppelbyte / 1. Zufallszahlen-Doppelelement der Ausgangs- und der aktuellen Zahl miteinander verglichen werden sollen. Die 2er-Potenz der Bitanzahl bildet den Wert, der angibt, wie viele Blöcke im Durchschnitt mit
- ein und demselben Operations-Schlüssel ('Safety'=0) bzw.
- denselben Komponenten ('Safety'=1,2,3)
behandelt werden dürfen, d.h. für 'Safety'=0 m u s s gelten:
MeanBlks_ := 2**q -1 mit q >= 0 u n d
mit q < 8*Gruppe - ln(Blockelemente/Schlüsselelemente) / ln(2) <= 16*Gruppe

5.2.13 MeanBlksX

=n - für Operationsschlüssel-Blöcke
=63,8388607,4294967295 - aktuelle Werte für CODING1/2/3

5.2.14 MeanBlksD

=n - für Daten-Blöcke
- <=2**6 –1 / <=2**23 –1 / <=2**32 –1
=63,8388607,4294967295 - aktuelle Werte für CODING1/2/3

5.2.15 MeanBlksO

=n - für Überladedaten-Blöcke
- <=2**6 –1 / <=2**23 –1 / <=2**32 –1
=63,8388607,4294967295 - aktuelle Werte für CODING1/2/3

6 Kontakt-Daten

eMail-Adresse:
sysjm@t-online.de, Betreff "Coding"