Universität Karlsruhe (TH)
Forschungsuniversität · gegründet 1825

Seitenkanalresistente symmetrische Verschlüsselung für digitale Tachographen

Diplomarbeit am Institut für Algorithmen und Kognitive Systeme
Fakultät für Informatik
Universität Karlsruhe (TH)

von

cand. inform.
Nora Lieberknecht

Betreuer:
Prof. Dr. Dejan Lazic
Dipl.-Inform. Jens-Matthias Bohli

Tag der Anmeldung: 1. September 2006
Tag der Abgabe: 28. Februar 2007

Ich erkläre hiermit, daß ich die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

Karlsruhe, den 27. Februar 2007

Inhaltsverzeichnis

Abbildungsverzeichnis

1. Einführung

Sicher ist, daß nichts sicher ist. Selbst das nicht.
Joachim Ringelnatz

In unserer heutigen, modernen Welt sind wir von immer mehr sichtbaren und unsichtbaren informationsverarbeitenden Systemen umgeben. Diese übernehmen Funktionen, die uns das Leben erleichtern und für steigenden Komfort sorgen. Sie kommen beispielsweise im elektronischen Zahlungsverkehr, im Automobil, in Mobiltelefonen und Fernsehgeräten zum Einsatz. Die ständig wachsende Zahl dieser Systeme und ihre zunehmende Vernetzung birgt jedoch auch Risiken und Gefahren hinsichtlich der verarbeiteten Daten und deren Übertragung. Diese sollen vor Manipulationen oder unerlaubten Zugriffen geschützt werden und nur dazu autorisierten Personen und vertrauenswürdigen Geräten zugänglich sein. Dazu werden kryptographische Methoden eingesetzt, die unter Verwendung geheimer Schlüssel die Sicherheit der Systeme sowie den Schutz der übertragenen Daten gewährleisten.

Dabei kann man im Allgemeinen zwischen symmetrischen und asymmetrischen Verfahren unterscheiden. Erstere verwenden zur Ver- und Entschlüsselung denselben Schlüssel, woraus sich das Problem des sicheren Austauschs dieses (geheimen) Schlüssels zwischen den Kommunikationspartnern ergibt. Sie werden hauptsächlich zur Ver- und Entschlüsselung von Daten eingesetzt und sind in ihrer Ausführung wesentlich schneller als die asymmetrischen Verfahren. Bei diesen wird das Problem der sicheren Schlüsselverteilung durch die Verwendung eines Schlüsselpaars, bestehend aus einem öffentlich bekannten und einem geheimen (privaten) Schlüssel gelöst. Sie basieren auf sogenannten Einwegfunktionen, bei denen eine Richtung sehr einfach, die umgekehrte jedoch extrem schwer bis unmöglich in adäquater Zeit zu berechnen ist. Asymmetrische Verfahren werden in erster Linie zur Signierung von Daten sowie für den sicheren Austausch geheimer symmetrischer Schlüssel eingesetzt.

Bei beiden Verfahren beruht die Sicherheit in erster Linie auf einer ausreichend großen Schlüssellänge der verwendeten geheimen Schlüssel, sodaß diese nicht durch Ausprobieren aller möglichen Schlüssel enthüllt werden können. Dies ist gerade in Zeiten wachsender Rechenleistung ein ernstzunehmendes Problem.

Seit Ende der 90er Jahre gibt es jedoch noch eine weitere wichtige und mächtige Angriffsform auf kryptographische Systeme, bei der die physikalischen Eigenschaften der Implementierung dieser Verfahren in einem Gerät oder einer Chipkarte für einen Angriff genutzt werden. Durch Messung dieser Eigenschaften kann ein physikalischer Kommunikationskanal, ein sogenannter *Seitenkanal* aufgebaut werden, aus dem Informationen über die intern verarbeiteten Daten und den dabei verwendeten geheimen Schlüssel gewonnen werden können. Dabei wird die mathematische Sicherheit eines Algorithmus umgangen, sodaß auf diese Weise auch als sicher geltende kryptographische Verfahren durch ihre praktische Umsetzung unsicher werden. Implementierungen dieser Algorithmen müssen daher durch entsprechende Gegenmaßnahmen vor dieser Form von Angriffen geschützt werden.

Hierfür gibt es verschiedene Möglichkeiten, die von der jeweiligen Implementierung sowie der gewählten Technologie abhängen. Sie wurden im Laufe dieser Arbeit detailliert untersucht und es wurde eine gegen Seitenkanalangriffe resistente Version einer Hardware-Implementierung des Algorithmus Triple-DES als Schaltung entwickelt.

Einsatzbereich dieser Schaltung sind die von der europäischen Union eingeführten digitalen Tachographen, die in Lastkraftwagen ab 3,5t und Bussen zur Aufzeichnung der gefahrenen Wegstrecke und Geschwindigkeit sowie zur Kontrolle der Lenk- und Ruhezeiten der Fahrer eingesetzt werden. Sie lösen die mechanischen Tachographen ab, die verhältnismäßig leicht zu manipulieren waren. Dadurch kam es häufig zu Unfällen, da die vorgeschriebenen Geschwindigkeiten sowie die Ruhezeiten der Fahrer oft nicht eingehalten wurden. Dies soll bei den digitalen Tachographen durch den Einsatz kryptographischer Verfahren verhindert werden. Die Manipulations- und Datensicherheit der Geräte soll damit erhöht und ein Beitrag zur Erhöhung der Verkehrssicherheit geleistet werden.

1.1 Zielsetzung der Arbeit

Ziel dieser Arbeit war die Entwicklung einer seitenkanalresistenten Implementierung des Data Encryption Standard (DES) bzw. Triple-DES als Schaltung in Hardware. Dabei sollte die von der europäischen Verordnung für die digitalen Tachographen geforderte Sicherheitsstufe erreicht werden, die unter anderem eine Resistenz gegen Seitenkanalangriffe verlangt. Auf diese Weise soll es unmöglich sein, die in den Komponenten des Tachographen gespeicherten geheimen Schlüssel durch Angriffe jedweder Art zu enthüllen.

Die Schaltung sollte anschließend synthetisiert und damit ein rekonfigurierbarer Chip, ein sogenannter FPGA[1], programmiert werden. Anhand von Seitenkanalmessungen an diesem Chips sollte dann die Resistenz gegen Seitenkanalangriffe verifiziert werden.

[1] Field Programmable Gate Array

1.2 Gliederung

Kapitel 2 gibt zunächst einen Überblick über die von der europäischen Gemeinschaft eingeführten digitalen Tachographen und deren gesetzliche Grundlage. Anschließend werden die einzelnen Komponenten des Tachographen sowie seine Funktionen dargestellt.

In Kapitel 3 werden Sicherheitsaspekte informationstechnischer Systeme sowie deren Bewertung nach einheitlichen Kriterien beschrieben. Weiterhin werden die für die einzelnen Komponenten des digitalen Tachographen von der europäischen Verordnung geforderten Sicherheitsstufen erläutert.

Kapitel 4 behandelt die Kommunikation zwischen dem Bewegungssensor, der die Impulse zur Geschwindigkeits- und Wegstreckenmessung liefert und der Fahrzeugeinheit, die diese Impulse empfängt. Um eine Manipulation durch Auslassen von Impulsen zu verhindern, muß diese Kommunikation mit kryptographischen Mitteln besonders geschützt werden.

Der dazu eingesetzte kryptographische Algorithmus ist Triple-DES, eine Weiterentwicklung des Data Encryption Standards (DES). Dessen Funktionsweise sowie der Ablauf der Ver- und Entschlüsselung werden in Kapitel 5 detailliert beschrieben.

Kapitel 6 beschäftigt sich mit den verschiedenen Implementierungsformen kryptographischer Verfahren sowie mit den möglichen Angriffsformen auf solche Systeme. Dabei werden besonders Implementierungsattacken und im speziellen Seitenkanalangriffe behandelt. Außerdem werden verschiedene Gegenmaßnahmen gegen diese Angriffe vorgestellt.

Das 7. Kapitel beschreibt den praktischen Teil dieser Diplomarbeit, in der eine seitenkanalresistente Schaltung des DES- bzw. Triple-DES-Algorithmus entwickelt wurde.

Das letzte Kapitel fasst die Arbeit zusammen und gibt einen Ausblick auf zukünftige Entwicklungen.

2. Digitale Tachographen

2.1 Einführung

Seit Anfang der 50er Jahre sind in Deutschland in Lastkraftwagen ab 3,5t und Bussen sogenannte *Tachographen* oder Fahrtenschreiber vorgeschrieben, um die gefahrene Wegstrecke und Geschwindigkeit aufzuzeichnen und die Lenk- und Ruhezeiten der Fahrer kontrollieren zu können. Die Aufzeichnung erfolgt dabei auf einer runden Papierscheibe, die sich in der Fahrzeugeinheit im Cockpit befindet und von einem Uhrwerk in 24 Stunden um 360° gedreht wird. Über der Scheibe befindet sich eine Schreibernadel, die sich vom Scheibenmittelpunkt umso weiter entfernt, je höher die Geschwindigkeit ist. Die Impulse zur Weg- und Geschwindigkeitsmessung liefert ein Impulsgeber, der im Getriebe des Fahrzeugs befestigt ist. Dies ist beispielsweise ein Halleffektsensor, der besonders zur berührungslosen Drehzahlerfassung geeignet ist. Er ist über ein Kabel mit der Fahrzeugeinheit verbunden. Vor dem Sensor befindet sich ein Zahnrad, dessen Umdrehung proportional zur gefahrenen Wegstrecke und Geschwindigkeit ist. Jedesmal wenn sich ein Zahn am Sensor vorbeibewegt, sendet dieser einen Impuls an die Fahrzeugeinheit. Da die Übertragung der Impulse über ein ungeschütztes Kabel erfolgt, kann sie sehr leicht manipuliert werden, indem man das Kabel gegen eines austauscht, das weniger Impulse überträgt. Weitere Manipulationsformen der mechanischen Tachographen sind verbogene Zeiger, Wegbegrenzungen durch Gummi oder Schaumstoffteile und Kurzschlussschaltungen bis hin zum „Vergessen des Einlegens“ der Scheibe bei Fahrtbeginn oder unerlaubte Tachoscheibenwechsel. Dabei gibt es einen regelrechten „Industriezweig“ für Tachofälschungen [Blum03].

Dadurch kommt es häufig zu schweren Unfällen durch übermüdete Fahrer, die sich nicht an die gesetzlich vorgeschriebenen Lenk- und Ruhezeiten halten und am Steuer ihres Fahrzeugs einschlafen. Oft stehen die Fahrer unter permanentem Termindruck und der Konkurrenzkampf der Transportunternehmen sowie deren kommerzielle Interessen kollidieren mit den gesetzlichen Regelungen. Hinzu kommt, daß in den verschiedenen Ländern der europäischen Union unterschiedliche Gesetze und Regelungen gelten, was zu einer Benachteiligung einzelner Länder und zu einem unfairen Wettbewerb der Transportunternehmen führt.

Aus diesem Grund und um die Manipulation der Tachographen zu verhindern und damit die Verkehrssicherheit zu erhöhen, hat das EU-Parlament eine Verordnung erlassen, nach der ab dem 1. Mai 2006 alle neu zugelassenen LKW mit *digitalen Tachographen* ausgestattet werden müssen. Mit der Verordnung wird eine weitgehende Harmonisierung der Arbeits- und Ruhezeitenregelung, sowie der Verkehrs- und Unternehmenskontrollen angestrebt. Außerdem soll die Schaffung eines fairen Wettbewerbs im internationalen Transportverkehr unterstützt werden, da der digitale Tachograph in allen Mitgliedstaaten der europäischen Union verpflichtend eingeführt wird[1]. Da der digitale Tachograph und dessen Komponenten einem hohen Sicherheitsstandard genügen müssen, wird so ein Beitrag zur Erhöhung der Manipulations- und Datensicherheit geleistet. Die Sicherheit wird unter anderem dadurch gewährleistet, daß das Signal des Impulsgebers am Getriebeausgang mit kryptographischen Verfahren verschlüsselt wird, sodaß eine Manipulation der Zeit- oder Wegstreckenmessung durch „Verschlucken“ von Impulsen nicht mehr möglich ist.

2.2 Gesetzliche Grundlage

Am 11. April 2006 trat die EG-Verordnung 561/2006[2] zur Neuregelung der Lenk- und Ruhezeiten für Berufskraftfahrer in Kraft. Danach müssen in allen EU-Ländern ab dem 1. Mai 2006 alle neu zugelassenen Fahrzeuge ab einem höchstzulässigen Gesamtgewicht von 3,5t und Busse mit mehr als 9 Sitzplätzen mit einem digitalen Tachographen ausgerüstet sein.

Der Starttermin war ursprünglich für den 05. August 2004 vorgesehen, dies konnte allerdings nicht realisiert werden, da es an entsprechenden praxistauglichen digitalen Kontrollgeräten sowie an entsprechenden Bauartgenehmigungen hierfür fehlte. Daher konnten die Mitgliedstaaten auch keine passenden Fahrerkarten ausgeben. Außerdem wollte die europäische Kraftfahrzeugindustrie die digitalen Kontrollgeräte wegen ihrer technischen Komplexität[3] mindestens ein Jahr lang testen. Der Termin war daraufhin im Zuge einer Duldungsregelung zunächst bis zum 5. August 2005, später bis zum 1. Januar 2006 verlängert worden.

Bereits seit dem 5. August 2005 ist in Deutschland der Einbau auf freiwilliger Basis zugelassen. Es stand dem Unternehmer frei, ein Fahrzeug mit einem analogen oder digitalen Tachographen auch nach dem 5. August 2005 zuzulassen oder einzusetzen. Ab dem 1. Mai 2006 schließlich muss jedes Unternehmen seine Neufahrzeuge mit einem digitalen Tachographen ausrüsten[4]. In insgesamt 29 Staaten[5] ist der digitale Tachograph verpflichtend. Dadurch soll unter anderem die Wettbewerbsfähigkeit der Transportunternehmen der verschiedenen Länder sichergestellt werden, da nun in allen Ländern für die Fahrer die gleichen Lenk- und Ruhezeiten gelten. Außerdem

[1] http://www.asfinag.at/index.php?idtopic=240

[2] Verordnung (EG) Nr. 561/2006 des europäischen Parlaments und des Rates vom 15. März 2006 zur Harmonisierung bestimmter Sozialvorschriften im Straßenverkehr und zur Änderung der Verordnungen (EWG) Nr. 3821/85 und (EG) Nr. 2135/98 des Rates sowie zur Aufhebung der Verordnung (EWG) Nr. 3820/85 des Rates

[3] gemäß Verordnung (EG) Nr. 1360/2002 der Kommission vom 13. Juni 2002 zur siebten Anpassung der Verordnung (EWG) Nr. 3821/85 des Rates über das Kontrollgerät im Straßenverkehr an den technischen Fortschritt

[4] http://www.bsi.bund.de/zertifiz/index.htm

[5] dies sind die 25 EU-Mitgliedstaaten inklusive der neu hinzugekommenen sowie die 4 assoziierten Nachbarstaaten Norwegen, Schweiz, Liechtenstein und Island

wird mit den neuen digitalen Geräten eine Manipulation des Tachographen durch den Einsatz kryptographischer Verfahren und Protokolle zur Sicherung der Kommunikation zwischen Sensor und Fahrzeugeinheit im Vergleich zu den analogen Geräten erheblich erschwert.

2.3 Komponenten

Der digitale Tachograph besteht aus den in Abb. 2.1 dargestellten Komponenten. Die wichtigsten sind: das *digitale Kontrollgerät* - auch als *Fahrzeugeinheit* bezeichnet - der *Bewegungssensor*, der mit einem Getriebeteil verbunden ist und die Impulse zur Weg- und Geschwindigkeitsmessung an die Fahrzeugeinheit liefert, sowie die *Kontrollgerätkarten*. Das *Kombi-Instrument* in Abb. 2.1 entspricht im Wesentlichen einem Tachometer, wie er auch in Pkws zur Anzeige der Geschwindigkeit und der gefahrenen Wegstrecke verwendet wird.

Abbildung 2.1: Komponenten des digitalen Tachographen

2.3.1 Fahrzeugeinheit

Das *digitale Kontrollgerät* bzw. die *Fahrzeugeinheit* (engl. vehicle unit, VU), ist fest im Fahrzeug installiert und erfasst alle fahrzeug- sowie fahrerrelevanten Daten elektronisch. Ihre Aufgabe ist es, Daten über die Tätigkeit der Fahrer aufzuzeichnen, zu speichern, anzuzeigen, auszudrucken und auszugeben. Sie ist an einen Weg- und/oder Geschwindigkeitsgeber angeschlossen, mit dem sie Daten über die Fahrzeugbewegung austauscht [Geme02]. Die Benutzer identifizieren sich gegenber der VU durch sogenannte *Kontrollgerätkarten*. Die VU zeichnet die Tätigkeitsdaten der Benutzer auf und legt sie in ihrem Massenspeicher ab. Die Benutzerdaten werden außerdem auf den Kontrollgerätkarten aufgezeichnet und an Anzeigegeräte, Drucker und externe Geräte ausgegeben.

Die Fahrzeugeinheit besteht im Einzelnen aus den folgenden Komponenten:

- Prozessor
- Massenspeicher (für 365 Tage fahrer- und fahrzeugbezogene Daten)
- Echtzeituhr
- Drucker (für definierte Ausdrucke zu Kontroll- und Informationszwecken)
- 2 Chipkartenschnittstellen
- Display
- Optische Warneinrichtung
- Schnittstellen für Download und Programmierung.

Der gesamte digitale Tachograph - und damit insbesondere die Fahrzeugeinheit und der Bewegungssensor - müssen unter allen gewöhnlich im Gebiet der europäischen Gemeinschaft auftretenden klimatischen Bedingungen korrekt funktionieren. So muß er beispielsweise in einem Temperaturbereich von -20 bis +70° C fehlerfrei arbeiten können. Der Inhalt des Datenspeichers muß sogar bis zu einer Temperatur von -40° C erhalten bleiben. Die Fahrzeugeinheit besitzt verschiedene Kanäle, um die aufgezeichneten Daten auszugeben. Sie können auf dem Display angezeigt oder ausgedruckt werden, außerdem gibt es einen sicheren Kanal für den Download der Daten.

Das Kontrollgerät verfügt über vier Betriebsarten:

- Betrieb
- Kontrolle
- Kalibrierung
- Unternehmen.

Je nachdem welche gültigen Kontrollgerätkarten in die Kartenschnittstellen der Fahrzeugeinheit eingesteckt sind, schaltet das Kontrollgerät auf eine dieser Betriebsarten. Der Zugriff auf die gespeicherten Daten hängt von der jeweiligen Betriebsart ab und wird durch die Funktionalität der Zugriffskontrolle des Kontrollgerätes gewährleistet. Diese entscheidet darüber, ob der Zugriff auf die verschiedenen Ressourcen gewährt oder verweigert wird. Ungültige Karten, die eingesteckt werden, sind vom Kontrollgerät zu ignorieren, jedoch sollen das Anzeigen, Ausdrucken oder Herunterladen von auf abgelaufenen Karten gespeicherten Daten möglich sein [Geme02].

2.3.2 Bewegungssensor

Der *Bewegungssensor* (engl. motion sensor, MS) des digitalen Tachographen dient als Weg- und/oder Geschwindigkeitsgeber und ist zum Einbau in Straßentransportfahrzeuge vorgesehen [Geme02]. Seine Aufgabe ist es, der Fahrzeugeinheit gesicherte Daten im Hinblick auf die Fahrzeuggeschwindigkeit und die zurückgelegte Wegstrecke zur Verfügung zu stellen. Er ist mit einem Fahrzeugteil, dessen Bewegung proportional zur Fahrtgeschwindigkeit bzw. der zurückgelegten Wegstrecke ist, mechanisch verbunden. Er kann im Getriebe oder an einer anderen Stelle im Fahrzeug installiert werden. Der Sensor besteht aus einem Impulsrad und einem Richtungsanzeiger, der die Drehrichtung des Rades erkennt. Dadurch erhält er Informationen darüber, ob das Fahrzeug vorwärts oder rückwärts fährt und dementsprechend kann die zurückgelegte Wegstrecke berechnet werden.

Der Sensor besitzt außerdem einen 16 Bit Zähler, der mit jedem Impuls des Signals dekrementiert wird. Der Zähler kann also maximal $2^{16} = 65536$ verschiedene Werte besitzen. Auf eine bestimmte Instruktion der Fahrzeugeinheit hin wird der aktuelle Wert des Zählers eingefangen und verschlüsselt an die Fahrzeugeinheit gesendet. Diese besitzt ebenfalls einen solchen Zähler und vergleicht den vom Sensor erhaltenen Wert mit dem eigenen Zählerstand. Bei Übereinstimmung kann sie davon ausgehen, daß keine Impulse verschluckt oder hinzugefügt wurden und die Anzahl der Impulse und somit auch die gemessene Wegstrecke und Geschwindigkeit korrekt übertragen wurden.

Als Bewegungssensoren können beispielsweise *Hall-Sensoren* eingesetzt werden, da diese für die berührungslose Drehzahlerfassung an Zahnrädern besonders geeignet sind[6]. Der Sensor detektiert die Bewegung von ferromagnetischen Strukturen - wie zum Beispiel Zahnrädern - über die Veränderung des magnetischen Flusses. Das Sensorelement ist mit einem Permanentmagneten vorgespannt. Ein Zahn oder eine Lücke, die sich am Sensor vorbei bewegt, beeinflussen das Magnetfeld unterschiedlich. Dadurch wird bei einem Hallsensor eine Änderung der Hallspannung erzeugt, die in eine elektrische Größe umgesetzt und entsprechend gefiltert und aufbereitet werden kann.

Da die vom Bewegungssensor gelieferten Impulse proportional zur gefahrenen Wegstrecke und der Geschwindigkeit sind, konzentrieren sich Manipulationsversuche häufig auf diesen Teil des digitalen Tachographen. Die möglichen Sicherheitsgefährdungen des Sensors können unterteilt werden in „Sicherheitsgefährdungen im Zusammenhang mit der Zugriffskontrolle“, „Konstruktionsbedingte Sicherheitsgefährdungen“ und „Betriebsbedingte Sicherheitsgefährdungen“ [PfGr05]. In die erste Kategorie fallen Versuche seitens der Benutzer, Zugriff auf ihnen nicht erlaubte Funktionen zu erlangen. Beispiele für konstruktionsbedingte Sicherheitsgefährdungen des Sensors sind Fehler bei Hardware, Software oder Kommunikationsverfahren, die den Impulsgeber in einen unvorhergesehenen Zustand versetzen, der seine Sicherheit beeinträchtigt oder Versuche seitens der Benutzer, auf illegale Weise (aus Unterlagen des Herstellers oder durch Methoden des *Reverse Engineering*) Kenntnis über Konstruktionsdaten zu erlangen. Zu den betriebsbedingten Sicherheitsgefährdungen gehören schließlich Einwirkungen auf den Sensor von außen (thermisch, elektromagnetisch, optisch, chemisch, mechanisch, usw.) sowie Versuche, Änderungen an der Hardware

[6] http://www.rheintacho.de/produkte/sensoren/technik.htm

oder der Software vorzunehmen, die Eingabe oder die Daten des Impulsgebers zu manipulieren oder auf illegale Weise Kenntnis über Sicherheitsdaten während deren Generierung, Übertragung bzw. Speicherung im Gerät zu erlangen. Aufgrund dieser Vielzahl möglicher Angriffsszenarien muß die Kommunikation zwischen Sensor und Fahrzeugeinheit besonders geschützt werden. Der Ablauf dieser Kommunikation wird in Kapitel 4 genauer beschrieben.

2.3.3 Kontrollgerätkarten

Ein weiterer wichtiger Teil des neuen digitalen Tachographen sind die *Kontrollgerätkarten*. Sie bestehen aus einem Plastikkörper, auf dem optische Sicherheitsmerkmale und ein Mikrocontroller aufgebracht sind. Dieser überwacht die Datenspeicherung und die Zugriffsberechtigung. Die Chipkarten sind mit eigenen individuellen Schlüsseln und Zertifikaten versehen und ermöglichen einen gesicherten Zugriff auf Daten und Funktionen des digitalen Kontrollgerätes.

Eine weitere grundlegende Funktion der Kontrollgerätkarten ist das Speichern der Karten- und Karteninhaberkenndaten. Diese werden von der Fahrzeugeinheit verwendet um den Karteninhaber zu identifizieren, dementsprechende Funktionen und Datenzugriffsrechte zu gewähren und sicherzustellen, daß dem Karteninhaber seine Tätigkeiten zugerechnet werden können. Außerdem werden auf den Karten auch die Karteninhabertätigkeitsdaten, Ereignisse und Störungen sowie Kontrolltätigkeitsdaten gespeichert.

Die Karten werden in Deutschland vom *Kraftfahrt-Bundesamt (KBA)* mit den individuellen Personen- oder Firmendaten versehen und versendet. Die ausgegebenen Chipkarten werden in einer Datenbank, dem sogenannten *Zentralen Kontrollgerätkartenregister (ZKR)* registriert, das ebenfalls vom Kraftfahrt-Bundesamt geführt wird und mit den entsprechenden Stellen anderer europäischer Länder verbunden ist.

Insgesamt gibt es vier verschiedene Kontrollgerätkarten[7]:

- Die *Fahrerkarte* - ist eine dem Fahrer persönlich zugeordnete Kontrollgerätkarte, die mit dem Lichtbild und der Unterschrift des Fahrers versehen ist und anstelle der bisher verwendeten (analogen) Tachoscheibe benötigt wird. Musste der Fahrer bisher das Tachoblatt jeden Tag wechseln und die Blätter der letzten Woche mit sich führen, so erfolgen nunmehr alle Aufzeichnungen elektronisch und es werden bis zu 28 Tage auf der Fahrerkarte gespeichert. Somit sind sämtliche Fahreraktivitäten (Fahrt-, Arbeits- und Ruhezeiten, sowie Geschwindigkeit und andere KFZ-relevante Daten) auf der Karte gespeichert. Außerdem muß die Fahrerkarte verschiedene Kartenkenndaten speichern können. Dies sind die Kartenummer, der Name der ausstellenden Behörde sowie des ausstellenden Mitgliedsstaates, das Ausstellungsdatum sowie die Gültigkeitsdauer der Karte. Desweiteren enthält die Fahrerkarte die Karteninhaberdaten Name, Vorname, Geburtsdatum und Muttersprache des Fahrers, sowie dessen Führerscheinnummer. Die Fahrerkarte muß für jeden Kalendertag, an dem sie benutzt wurde, sowie für jeden Betriebszeitraum eines Fahrzeugs an diesem Tag die folgenden Daten speichern können [Geme02]:

[7] http://www.asfinag.at/index.php?idtopic=248

  - Datum und Uhrzeit der ersten Einsatzes des Fahrzeugs
  - Kilometerstand zu diesem Zeitpunkt
  - Datum und Uhrzeit des letzten Einsatzes des Fahrzeugs
  - Kilometerstand zu diesem Zeitpunkt
  - amtliches Kennzeichen und zulassender Mitgliedstaat.

  Außerdem dient die Fahrerkarte der Speicherung von Fahrertätigkeits- sowie Ereignisdaten. Zu den Fahrertätigkeitsdaten gehören unter anderem das Datum des Arbeitstages, die an diesem Tag zurückgelegte Gesamtwegstrecke, der Ort des Beginns und/oder Endes des Arbeitstages sowie der Kilometerstand. Zu den auf der Karte gespeicherten Ereignisdaten gehören die Unterbrechung der Stromversorgung, Datenfehler bei Weg und Geschwindigkeit sowie Versuche einer Sicherheitsverletzung. Für jedes Ereignis werden ein Ereigniscode, Datum und Uhrzeit des Ereignisbeginns und -endes, sowie das amtliche Kennzeichen und der zulassende Mitgliedstaat des Fahrzeugs, in dem das Ereignis auftrat, gespeichert. Die Gültigkeit der Fahrerkarte beträgt 5 Jahre.

  Nachdem eine gültige Fahrerkarte in die Fahrzeugeinheit eingesteckt wurde, authentifizieren sich beide gegenseitig. Dabei prüft die Fahrzeugeinheit die Echtheit der Karte. Dies soll verhindern, daß Daten wie z.B. Lenkzeiten auf nicht autorisierte Karten geschrieben und damit nicht auf das Konto des Fahrers gebucht werden. Umgekehrt prüft die Fahrerkarte bei diesem Vorgang auch die Echtheit der Fahrzeugeinheit, da nur authentische Fahrzeugeinheiten berechtigt sind, Daten in der Fahrerkarte zu verändern.

Neben der Fahrerkarte gibt es noch drei weitere Arten von Kontrollgerätkarten:

- Die *Unternehmenskarte* - mit ihr können die Daten auf einem Kontrollgerät vor dem Zugriff durch andere Unternehmen geschützt werden. Dies empfiehlt sich z.B. vor allem dann, wenn mehrere Unternehmen das gleiche Fahrzeug nutzen. Die Unternehmenskarte ermöglicht das gesetzlich spätestens alle 28 Tage vorgeschriebene Herunterladen der Daten von den Fahrerkarten der Lenker und spätestens alle 3 Monate das Herunterladen der Daten aus dem digitalen Kontrollgerät. Die Unternehmenskarte wird auf Unternehmen, die ein gewerblich genutztes Fahrzeug mit inländischen Kennzeichen ausgerüstet mit einem digitalen Kontrollgerät einsetzen ausgestellt und ist für 5 Jahre gültig.

- Die *Werkstattkarte* - dient geschulten Personen in ermächtigten Werkstätten zur Aktivierung, Kalibrierung und Prüfung des digitalen Kontrollgerätes und berechtigt zum Herunterladen der Daten. Die Werkstattkarte ist mit einer PIN ausgestattet und wird persönlich auf einen Mechaniker ausgestellt, wobei die Gültigkeit 1 Jahr beträgt.

- Die *Kontrollkarte* - erlaubt das Auslesen der Daten aus dem Kontrollgerät durch Exekutiv- und Kontrollorgane und die Weiterverarbeitung der Daten mit elektronischen Hilfsmitteln. Sie wird an Personen mit verkehrsüberwachender Funktion (z.B. die Polizei) ausgegeben und dient unter anderem dazu, die Echtheit von Fahrzeugeinheiten zu überprüfen. Eine Kontrollkarte wird auf Kontrollstellen ausgestellt und ist für eine Dauer von 5 Jahren gültig.

2.4 Aufgaben

Der digitale Tachograph soll die folgenden Funktionen gewährleisten [Geme02]:

- Geschwindigkeits- und Wegstreckenmessung
- Zeitmessung
- Überwachung der Fahrertätigkeiten („Lenken“, „Arbeit“, „Bereitschaft“ und „Unterbrechung/Ruhe“)
- Überwachung des Status der Fahrzeugführung („Team“ oder „Einmannbetrieb“)
- Manuelle Eingaben durch die Fahrer (z.B. Ort des Beginns und/oder Endes des Arbeitstages)
- Unternehmenssperren
- Aufzeichnung und Speicherung von Daten im Massenspeicher
- Auslesen von Daten aus dem Massenspeicher
- Auslesen von Daten aus den Kontrollgerätkarten
- Aufzeichnung und Speicherung von Daten in den Kontrollgerätkarten
- Herunterladen von Daten auf externe Datenträger
- Datenausgabe an externe Zusatzgeräte
- Überwachung des Einsteckens und Entnehmens der Karten
- Überwachung von Kontrollaktivitäten („Anzeige“, „Druck“, „Fahrzeugeinheit“, „Herunterladen“)
- Feststellung von Ereignissen und Störungen
- Integrierte Tests und Selbsttests
- Kalibrierung
- Anzeige
- Ausdrucken
- Zeiteinstellung
- Warnung

Die Geschwindigkeitsmessung erfolgt auf mindestens 1 km/h genau in einem Bereich von 0 bis 220 km/h. Dabei muß die Geschwindigkeit „innerhalb der zulässigen Fehlergrenzen innerhalb von 2 Sekunden nach Abschluß einer Geschwindigkeitsänderung korrekt gemessen werden, wenn sich die Geschwindigkeit mit bis zu 2 m/s^2 geändert hat“ [Geme02]. Außerdem soll die Geschwindigkeitsmessung Informationen darüber enthalten, ob das Fahrzeug steht oder fährt. Letzteres wird angenommen, falls für

einen Zeitraum von mindestens 5 Sekunden mehr als ein Impuls pro Sekunde geliefert wird.

Die Zeitmessfunktion läuft ständig und stellt Datum und Uhrzeit digital in *UTC* (Universal Time, Coordinated) bereit. Die Zeitmessung erfolgt dabei auf 1 Sekunde genau.

3. Sicherheitsaspekte

3.1 Bewertung der Sicherheit von IT-Systemen

Um einen sicheren Umgang mit Daten und informationsverarbeitenden Systemen zu gewährleisten, ist es erforderlich, der jeweiligen Gefährdungslage entsprechende Sicherheitsstandards zu entwickeln und einzuhalten. Als einheitlicher Maßstab zur Beurteilung der Sicherheit informationstechnischer Systeme dienen *Kriterien* für die Prüfung und Bewertung der IT-Sicherheit[1].

Um Vertrauen in die Informationstechnik und Transparenz hinsichtlich der Sicherheitseigenschaften von IT-Produkten zu schaffen, wird die Prüfung und Bewertung von IT-Produkten und -Systemen nach solchen einheitlichen Kriterien durch unabhängige akkreditierte Prüfstellen vorgenommen. In Deutschland ist das *Bundesamt für Sicherheit in der Informationstechnik (BSI)* maßgeblich an der Erarbeitung dieser Sicherheitskriterien beteiligt. Eingebunden in viele Aktivitäten zur Erhöhung der IT-Sicherheit bietet das BSI die Dienstleistung der *Zertifizierung* von IT-Produkten und IT-Systemen im Hinblick auf deren Sicherheitseigenschaften an.[2]

Ein internationaler Standard für Kriterien zur Bewertung und Zertifizierung der Sicherheit von Computersystemen im Hinblick auf Datensicherheit und Datenschutz sind die *Common Criteria for Information Technology Security Evaluation* (kurz *Common Criteria*). Dieser Standard soll eine gemeinsame Grundlage für solche Bewertungen bieten und vermeiden, daß Komponenten oder Systeme in verschiedenen Ländern mehrfach zertifiziert werden müssen. Damit löst er den europäischen Standard *ITSEC (Information Technology Security Evaluation Criteria)*, den amerikanischen *TCSEC*-Standard[3] *(Trusted Computer System Evaluation Criteria)*, das sogenannte *Orange-Book*, sowie die kanadischen Kriterien *Canadian Trusted Computer Product Evaluation Criteria (CTCPEC)* ab.

Bei der Bewertung der Qualität (Vertrauenswürdigkeit) eines Computersystems wird zwischen der *Wirksamkeit* der Methode und der *Korrektheit* der Implementierung

[1] http://www.bsi.de/literat/faltbl/itsikrit.htm

[2] http://www.bsi.bund.de/zertifiz/index.htm

[3] ein von der US-Regierung herausgegebener Standard für die Bewertung und Zertifizierung der Sicherheit von Computersystemen

unterschieden. Die *Wirksamkeit* bezeichnet dabei die Widerstandsfähigkeit eines Schutzmechanismus gegen Umgehungsversuche, während bei der *Korrektheit* insbesondere auf Programmfehler geprüft wird, sowie darauf, inwieweit die Implementierung tatsächlich die zuvor bewertete Methode realisiert.

Die Common Criteria definieren sieben *Stufen der Vertrauenswürdigkeit* (engl. *Evaluation Assurance Level*) EAL 1-7, die die Korrektheit der Implementierung des betrachteten Systems bzw. die Prüftiefe beschreiben. Mit wachsender EAL-Nummer steigen dabei die Anforderungen an den zu prüfenden Umfang, die Prüftiefe und die Prüfmethoden. Auch bei ITSEC werden zur Bewertung des Vertrauens in die Korrektheit sechs Evaluationsstufen E1 bis E6 definiert. E1 bezeichnet dabei die niedrigste, E6 die höchste Stufe. Bei der Wirksamkeit erfolgt die Bewertung der Stärke der Mechanismen nach *niedrig*, *mittel* und *hoch*.

3.2 Sicherheitsanforderungen an digitale Tachographen

Da die Schlüssel im Speicher der Tachograph-Hardware sicher abgespeichert sind, muss die Tachograph-Hardware nach „ITSEC E3/Hoch„ zertifiziert sein, sodaß kein Auslesen bzw. Verändern von Schlüsseln möglich ist.

Die von der europäischen Verordnung für die digitalen Tachographen geforderte Stufe der Vertrauenswürdigkeit nach Common Criteria ist EAL 4+. Das bedeutet, daß das System „methodisch entwickelt, getestet und durchgesehen“[4] sein muß. Bei ITSEC entspricht dies der Stufe E3. Dies umfasst die „Bereitstellung von Quellcode bzw. Hardware-Konstruktionszeichnungen und Abbildung auf die Basiskomponenten“[5]. Bei der Wirksamkeit wird die Stufe *hoch* gefordert. Dies bedeutet: Mechanismen können nur von Angreifern überwunden werden, die über sehr gute Fachkenntnisse, Gelegenheiten und Betriebsmittel verfügen, wobei ein solcher erfolgreicher Angriff normalerweise als nicht durchführbar beurteilt wird.

Um die Kompatibiliät von Fahrzeugeinheiten und Tachographkarten verschiedener Hersteller zu gewährleisten und jedem autorisierten Kontrolleur die Inspektion der von jeder beliebigen Fahrzeugeinheit heruntergeladenen Daten zu ermöglichen, müssen gemeinsame Sicherheitsmechanismen und -anforderungen entwickelt werden.

Diese Sicherheitsanforderungen an das digitale Tachographsystem umfassen im Einzelnen die folgenden Punkte [LePW06]:

- Schutz der aufgezeichneten und gespeicherten Daten vor unautorisierten Zugriffen und Manipulationen sowie Erkennung von Versuchen dieser Art
- Wechselseitige Authentifizierung zwischen Fahrzeugeinheit und Kontrollgerätkarten
- Integrität und Authentizität der zwischen dem Bewegungssensor und der Fahrzeugeinheit ausgetauschten Daten

[4] http://www.commoncriteria.de
[5] http://www.bsi.de/zertifiz/itkrit/itsec.htm

- Integrität und Authentizität der zwischen der Fahrzeugeinheit und den Kontrollgerätkarten ausgetauschten Daten
- Integrität und Authentizität der heruntergeladenen Daten.

3.2.1 Anforderungen an die Kontrollgerätkarten

Die Sicherheitsanforderungen an die Tachographkarten beinhalten:

- Schutz der Integrität und Authentizität der zwischen den Karten und der Fahrzeugeinheit ausgetauschten Daten
- Schutz der Integrität und Authentizität der von den Karten heruntergeladenen Daten
- Ausschluß jeglicher Falsifizierungsmöglichkeit der in den Karten gespeicherten Daten
- Erkennen und Verhindern jeglicher Versuche dieser Art.

Der geforderte *Evaluation Assurance Level (EAL)* für die Tachographkarten ist ebenfalls EAL 4+ (Common Criteria) bzw. E3 (ITSEC).

3.2.2 Anforderungen an den Bewegungssensor

Die Sicherheitsanforderungen an den Bewegungssensor bzw. den Weg- und/oder Geschwindigkeitsgeber beinhalten unter anderem die folgenden Funktionen (Anlage 10 von [Geme02]):

- Für jede Interaktion Feststellung der Identität der angeschlossenen Geräteeinheit
- Authentisierung jeder angeschlossenen Fahrzeugeinheit (VU) bzw. jedes angeschlossenen Verwaltungsgeräts bei Anschließen der Geräteeinheit sowie bei Wiederzuschalten der Stromversorgung
- Wiederholung der Authentisierung der angeschlossenen VU in bestimmten Abständen
- Erkennen und Verhindern des Gebrauchs kopierter und wieder eingespielter Authentisierungsdaten
- Kontrolle der Zugriffsberechtigung auf Funktionen und Daten
- Durchsetzen geeigneter Zugriffsrechte für das Lesen und Schreiben von Sicherheitsdaten
- Ausgabe von Zuordnungsdaten auf Verlangen an authentisierte Geräteeinheiten

Nach Erkennen einer (vom Hersteller noch festzulegenden, jedoch 20 nicht übersteigenden) Zahl von aufeinanderfolgenden erfolglosen Authentisierungsversuchen wird die sicherheitserzwingende Funktion

- ein Auditprotokoll über das Ereignis anlegen,
- eine Warnung an die Geräteeinheit ausgeben und
- die Ausgabe von Weg- und Geschwindigkeitsdaten im ungesicherten Modus fortsetzen.

Der Weg- und/oder Geschwindigkeitsgeber legt bei Ereignissen, die seine Sicherheit beeinträchtigen, *Auditprotokolle* für die betreffenden Ereignisse an. Solche Ereignisse sind unter anderem fehlgeschlagene Authentisierungen, Integritätsfehler der Speicherdaten, unberechtigtes Öffnen des Gehäuses oder Hardwaremanipulation. Die Auditprotokolle enthalten die folgenden Angaben:

- Datum und Uhrzeit des Ereignisses
- Art des Ereignisses
- Identität der angeschlossenen Geräteeinheit.

Stehen die geforderten Daten nicht zur Verfügung, wird ein entsprechender Fehlervermerk ausgegeben. Der Weg- und/oder Geschwindigkeitsgeber überträgt die angefertigten Auditprotokolle zum Zeitpunkt ihrer Generierung an die Fahrzeugeinheit und kann sie zugleich in seinem Speicher ablegen. Für den Fall, daß der Weg- und/oder Geschwindigkeitsgeber Auditprotokolle speichert, muß sichergestellt sein, daß unabhängig von der anderweitigen Speicherbelegung 20 Auditprotokolle gespeichert und diese gespeicherten Auditprotokolle auf Anfrage an authentisierte Geräteeinheiten ausgegeben werden können.

Falls der Weg-/Geschwindigkeitsgeber physisch getrennte Teile nutzt und Daten zwischen diesen Teilen übertragen werden, müssen diese Daten gegen Verfälschungen geschützt werden. Desweiteren muß jedes Öffnen des Gehäuses festgestellt werden können, selbst wenn die externe Stromversorgung bis zu 6 Monate unterbrochen ist. Außerdem sollen Versuche der physischen Manipulation am Gerät erkannt werden. In Bezug auf den Datenbestand des Weg-/Geschwindigkeitsgebers soll sichergestellt sein, daß bei Bedarf auf die Daten zugegriffen werden kann und daß die Daten weder unnötig abgerufen noch zurückgehalten werden. Die Weg- und Geschwindigkeitsdaten werden mit den zugehörigen Sicherheitsattributen an die Fahrzeugeinheit übertragen, so daß diese in die Lage versetzt wird, die Integrität und Authentizität der Daten festzustellen.

Die Mindestrobustheit der Sicherheitsmechanismen des Weg-/Geschwindigkeitsgebers gemäß Definition in ITSEC ist *Hoch*.

3.3 Erzeugung und Verteilung der benötigten Schlüssel

Um die von der europäischen Verordnung für die digitalen Tachographen geforderte Sicherheitsstufe zu erreichen, werden verschiedene kryptographische Verfahren und entsprechende Schlüssel benötigt, sodaß Mechanismen für die Datenintegrität, die Authentifizierung, die digitale Signatur und die Vertraulichkeit des Datenaustauschs zur Verfügung gestellt werden. Die hierfür benötigten kryptographischen Verfahren sowie die Erzeugung und Verteilung der entsprechenden Schlüssel werden im Folgenden beschrieben.

3.3.1 RSA

Für die Authentisierung zwischen Fahrzeugeinheit und Kontrollgerätkarten sowie die sichere Übertragung von Sitzungsschlüsseln und die digitale Signatur von Daten, die von Fahrzeugeinheiten oder Kontrollgerätkarten an externe Medien heruntergeladen werden, wird das klassische *RSA-Public-Key*-Verfahren verwendet.

RSA[6] gehört zu den asymmetrischen Verschlüsselungsalgorithmen. Bei diesen Verfahren wird zur Verschlüsselung ein anderer Schlüssel als zur Entschlüsselung verwendet. Ein RSA-Schlüssel ist somit immer ein Schlüsselpaar, bestehend aus einem öffentlichen Schlüssel (engl. public key) für die Verschlüsselung und einem privaten Schlüssel (engl. private key) für die Entschlüsselung. Nur der private Schlüssel ist vom Besitzer geheim zu halten, der öffentliche ist im Allgemeinen öffentlich bekannt und für jedermann zugänglich.

Um zu garantieren, daß ein öffentlicher Schlüssel auch tatsächlich zu dessen Besitzer gehört und den Einsatz falscher (z.B. untergeschobener) Schlüssel zu verhindern, wird ein Nachweis benötigt, daß der verwendete öffentliche Schlüssel auch zum designierten Empfänger der verschlüsselten Nachricht bzw. zum Sender einer elektronisch signierten Nachricht gehört. Diesen Nachweis stellt eine vertrauenswürdige Stelle in Form eines (digitalen) *Zertifikates* aus. Dies kann von öffentlichen Zertifizierungsstellen oder von Personen, sogenannten *Notaren*, vorgenommen werden. Zertifikate bestätigen die Zugehörigkeit eines kryptographischen Schlüssels zu einer Person, Firma, Institution oder Maschine. Dadurch können Authentizität, Vertraulichkeit und Integrität von Daten gegenüber Dritten garantiert werden.

Der RSA-Algorithmus wird durch folgende Beziehungen vollständig definiert (Anlage 11 von [Geme02]):

$$X.PK[m] = s = m^e \bmod n$$
$$X.SK[s] = m = s^d \bmod n.$$

Dabei ist $X.PK[m] = s$ die RSA-Chiffrierung einer Information m unter Verwendung des öffentlichen Schlüssels (**P**ublic **K**ey) e eines Benutzers X und $X.SK[s]$ die RSA-Dechiffrierung eines Chiffrats s mit dem privaten Schlüssel (**S**ecret **K**ey) d des Benutzers X. Der Modulus n ist das Produkt zweier (geheimer) Primzahlen p und q und ist ebenfalls öffentlich bekannt. Die Schlüssel haben dabei folgende Länge: Modulus n 1024 Bit, öffentlicher Exponent e maximal 64 Bit, privater Exponent d 1024 Bit.

[6] nach seinen Erfindern **R**ivest, **S**hamir und **A**dleman

Die Sicherheit des RSA-Algorithmus wie die aller asymmetrischen kryptographischen Algorithmen basiert auf sogenannten *Einwegfunktionen*. Bei diesen Funktionen ist die eine Richtung (in diesem Fall das Produkt zweier großer Primzahlen) leicht zu berechnen, die umgekehrte Richtung (in diesem Fall die Faktorisierung des Produkts) jedoch sehr schwer und praktisch unmöglich durchzuführen. Eine genaue Beschreibung des RSA-Algorithmus findet sich unter anderem in [JoKa03].

Das für die digitalen Tachographen verwendete asymmetrische Kryptosystem basiert auf einer Standard-*Public-Key-Infrastruktur (PKI)*. Dabei gibt es drei hierarchische Funktionsebenen (siehe Anlage 11 von [Geme02]):

- die europäische Ebene
- die Mitgliedstaatebene
- und die Geräteebene.

Abb. 3.1 zeigt die gesamte Hierarchie dieser Public-Key-Infrastruktur.

Auf europäischer Ebene wird ein einziges Schlüsselpaar (EUR.SK und EUR.PK erzeugt. Der europäische private Schlüssel EUR.SK wird zur Zertifizierung der öffentlichen Schlüssel der Mitgliedstaaten verwendet. Diese Aufgabe wird von einer Europäischen Zertifizierungsstelle, der *European Root Certification Authority (ERCA)* wahrgenommen, die der europäischen Kommission untersteht.

Auf Mitgliedstaatebene wird ein Mitgliedstaatschlüsselpaar (MS_i.SK und MS_i.PK) erzeugt. Öffentliche Mitgliedstaatschlüssel werden von der Europäischen Zertifizierungsstelle ERCA zertifiziert. Der private Mitgliedstaatschlüssel MS_i.SK wird für die Zertifizierung von öffentlichen Schlüsseln verwendet, die in Geräten (Fahrzeugeinheiten oder Kontrollgerätkarten) eingefügt sind. Über alle zertifizierten öffentlichen Schlüssel müssen Belege aufbewahrt werden, zusammen mit der Kennung des Geräts, für das sie bestimmt sind.

Jeder Mitgliedstaat etabliert seine eigene *Member State Authority (MSA)*, die meistens durch eine staatliche Autorität wie z.B. das Verkehrsministerium repräsentiert wird und verschiedene Aufgaben übernimmt. Eine dieser Aufgaben ist die einer *Member State Certification Authority (MSCA)*, die Zertifizierungsstelle des jeweiligen Mitgliedsstaates. In Deutschland wird dies vom *Kraftfahrt-Bundesamt*[7] übernommen, in dem hierfür eines der größten *Trustcenter*[8] Europas eingerichtet wurde. Die Aufgabe einer solchen Zertifizierungsstelle ist unter anderem die Generierung von Schlüsselpaaren, die Bereitstellung öffentlicher Schlüssel, die Bestätigung der Echtheit und Gültigkeit von Signaturen sowie das Führen von Sperrlisten für ungültige Signaturen. Ein Mitgliedstaat darf sein Schlüsselpaar in regelmäßigen Abständen ändern.

Auf Geräteebene wird ebenfalls ein einziges Schlüsselpaar (EQT_j.SK und EQT_j.PK) erzeugt und in jedes Gerät eingefügt. Die öffentlichen Geräteschlüssel werden von der MSCA des jeweiligen Mitgliedstaates zertifiziert. Dieses Schlüsselpaar wird zur Authentisierung zwischen Fahrzeugeinheit und Tachographkarten, für die digitale

[7] http://www.kba.de

[8] Zertifizierungsstelle, die gegenüber Dritten die öffentlichen Schlüssel ihrer Kunden bestätigt

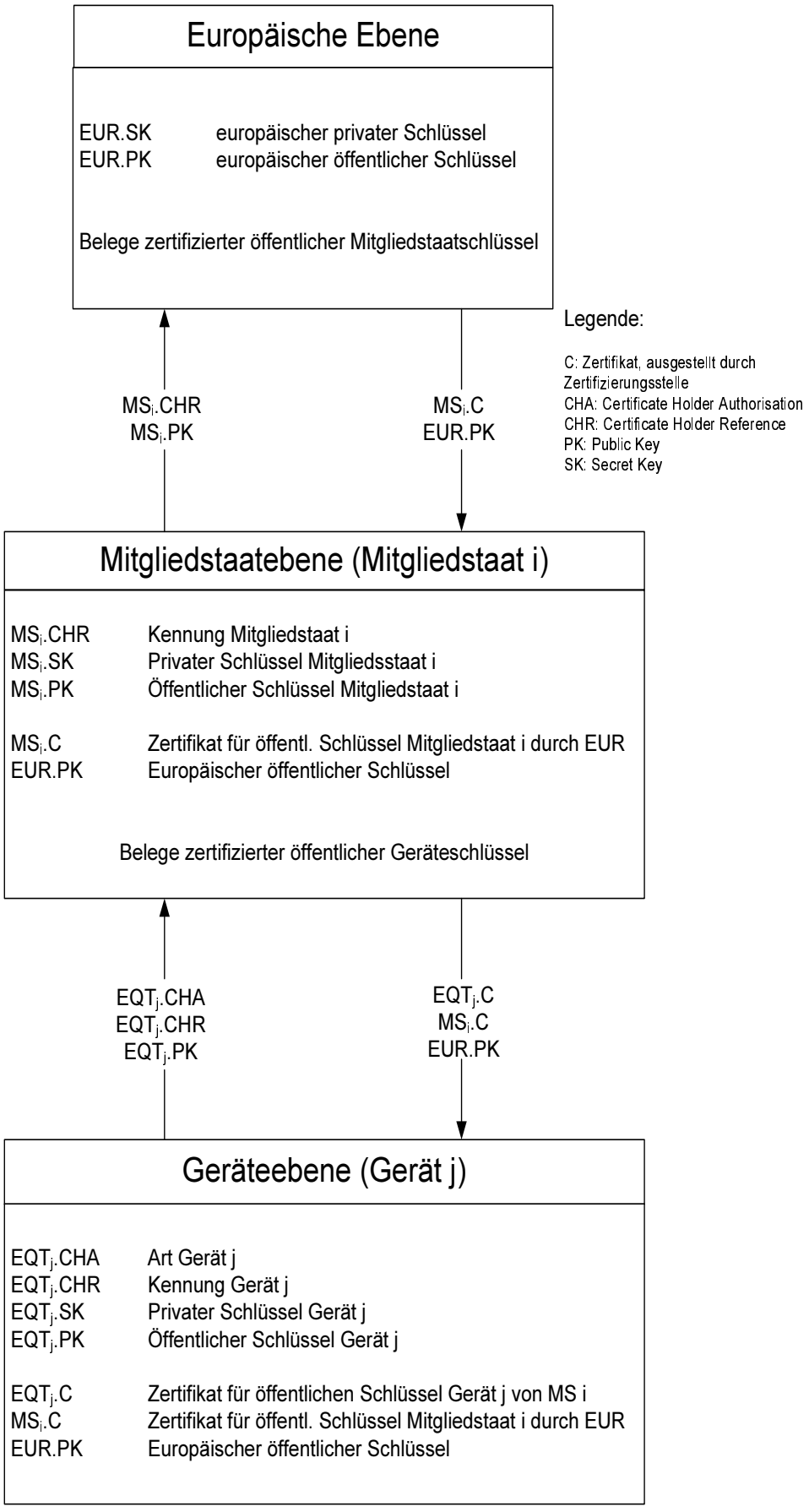

Abbildung 3.1: Hierarchie der Public-Key-Infrastruktur der digitalen Tachographen

Signatur der von der Fahrzeugeinheit oder den Tachographkarten heruntergeladenen Daten sowie für den sicheren Transport von Sitzungsschlüsseln zwischen der Fahrzeugeinheit und den Tachographkarten verwendet.

Bei der Erzeugung, der Übertragung sowie der Speicherung muß die Vertraulichkeit der privaten Schlüssel gewahrt werden. Dies stellt eine weitere Anforderung an die Hersteller der digitalen Tachographen dar.

3.3.2 Triple-DES

Um die Datenintegrität und die Vertraulichkeit der zwischen den einzelnen Komponenten des digitalen Tachographen ausgetauschten Daten zu gewährleisten, wird zusätzlich zum asymmetrischen RSA-Verfahren auch noch ein symmetrisches Verschlüsselungssystem verwendet.

Der bekannteste und weltweit am häufigsten eingesetzte symmetrische Algorithmus ist der 1977 vom US-amerikanischen *National Institute of Standards and Technology (NIST)* zum Standard erklärte *Data Encryption Standard*, kurz *DES*. Nachdem dieser im Jahr 1998 durch einen *Brute-Force*-Angriff[9] geknackt worden war, wurde *Triple-DES* verwendet, um die Schlüssellänge zu erhöhen und einen solchen Angriff aufgrund der derzeit verfügbaren Rechenleistung unmöglich zu machen.

Triple-DES besteht in der dreimaligen Hintereinanderausführung von DES, wobei es hinsichtlich der Anzahl der Schlüssel verschiedene Varianten gibt. Eine ist die Verwendung von zwei verschiedenen Schlüsseln. Sie wird auch mit *2TDES* bezeichnet [Nati99]. Dabei wird eine Nachricht m zunächst mit einem Schlüssel k_A verschlüsselt, anschließend mit einem zweiten Schlüssel k_B entschlüsselt und schließlich noch einmal mit k_A verschlüsselt. Triple-DES-Schlüssel haben somit die Form (k_A, k_B, k_A), wobei k_A und k_B unabhängige Schlüssel mit einer Länge von 64 Bit sind. Eine genaue Beschreibung von DES und Triple-DES findet sich in Kapitel 5 sowie in [Nati99].

Die europäische Zertifizierungsstelle ERCA generiert zwei voneinander unabhängige und einmalige Triple-DES-Schlüssel Km_{wc} und Km_{vu} mit einer Länge von jeweils 128 Bit. Die effektive Schlüssellänge beträgt somit jeweils 112 Bit, da 16 Bits als Paritätsbits dienen. Der erste Teilschlüssel, Km_{wc}, wird unter geeigneten Sicherheitsvorkehrungen zunächst an die Zertifizierungsstellen der Mitgliedstaaten (MS-CA) und von diesen an die Hersteller übermittelt und bei der Personalisierung der Karten in jede *Werkstattkarte* (engl. workshop card, WC) eingefügt. Der zweite Teilschlüssel, Km_{vu}, wird bei der Herstellung in jede Fahrzeugeinheit (engl. vehicle unit, VU) eingesetzt. Die beiden Teilschlüssel werden später zur Erzeugung des sogenannten *Masterkeys Km* verwendet, der für die Kommunikation zwischen Fahrzeugeinheit und Bewegungssensor benötigt wird. Diese wird im folgenden Kapitel genauer beschrieben.

[9] Ausprobieren aller möglichen Schlüssel

4. Kommunikation zwischen Fahrzeugeinheit und Bewegungssensor

Ziel der Einführung digitaler Tachographen war es unter anderem, eine Manipulation und damit die Verfälschung der gefahrenen Wegstrecke oder der Geschwindigkeit zu verhindern. Beide werden durch die Anzahl der Impulse bestimmt, die vom Bewegungssensor - dem Impulsgeber - zur Fahrzeugeinheit gesendet werden. Da die Übertragung dieser Impulse über eine ungesicherte Leitung erfolgt, die leicht manipuliert werden kann, werden kryptographische Methoden eingesetzt, um die übertragenen Daten zu verschlüsseln und so eine Verfälschung der übertragenen Impulse unmöglich zu machen. Die hierfür benötigten Schlüssel werden auf verschiedenen hierarchischen Ebenen erzeugt und verwaltet. Abb. 4.1 zeigt diese Verwaltung und Verteilung der Schlüssel, die zur symmetrischen Verschlüsselung der Kommunikation zwischen Fahrzeugeinheit und Bewegungssensor verwendet werden.

Die Fahrzeugeinheit enthält nach der Herstellung den 128 Bit langen Triple-DES-Teilschlüssel Km_{vu}, eine Seriennummer sowie einen konstanten Kontrollvektor CV. Der Sensor erhält eine erweiterte Seriennummer N_s. Diese ist 8 Byte groß und besteht aus der Seriennummer des Geräts (4 Byte), dem Herstellungsdatum (2 Byte), dem Sensortyp (1 Byte) sowie dem Namen des Herstellers (1 Byte). Dabei sind Hersteller, Typ und Seriennummer in der erweiterten Seriennummer verschlüsselt. Die erweiterte Seriennummer ist für jeden Sensor eindeutig. Außerdem erhält jeder Sensor bei der Herstellung einen individuellen sogenannten *Pairingkey* K_p. Dieser ist 128 Bit lang und ebenfalls für jeden Sensor einmalig. Er wird für das *Pairing* des Sensors mit einer Fahrzeugeinheit benötigt (vgl. Abschnitt 4.2).

N_s und K_p werden anschließend von der Member State Authority (MSA) an die MSCA, die Zertifizierungsstelle des jeweiligen Mitgliedstaates übermittelt. Diese besitzt einen *Masterkey* Km, der sich als XOR-Verknüpfung der beiden Teilschlüssel Km_{vu} und Km_{wc} berechnet sowie einen *Identifikationsschlüssel* K_{id}, der wiederum die XOR-Verknüpfung von Km und einem konstanten Kontrollvektor CV ist.

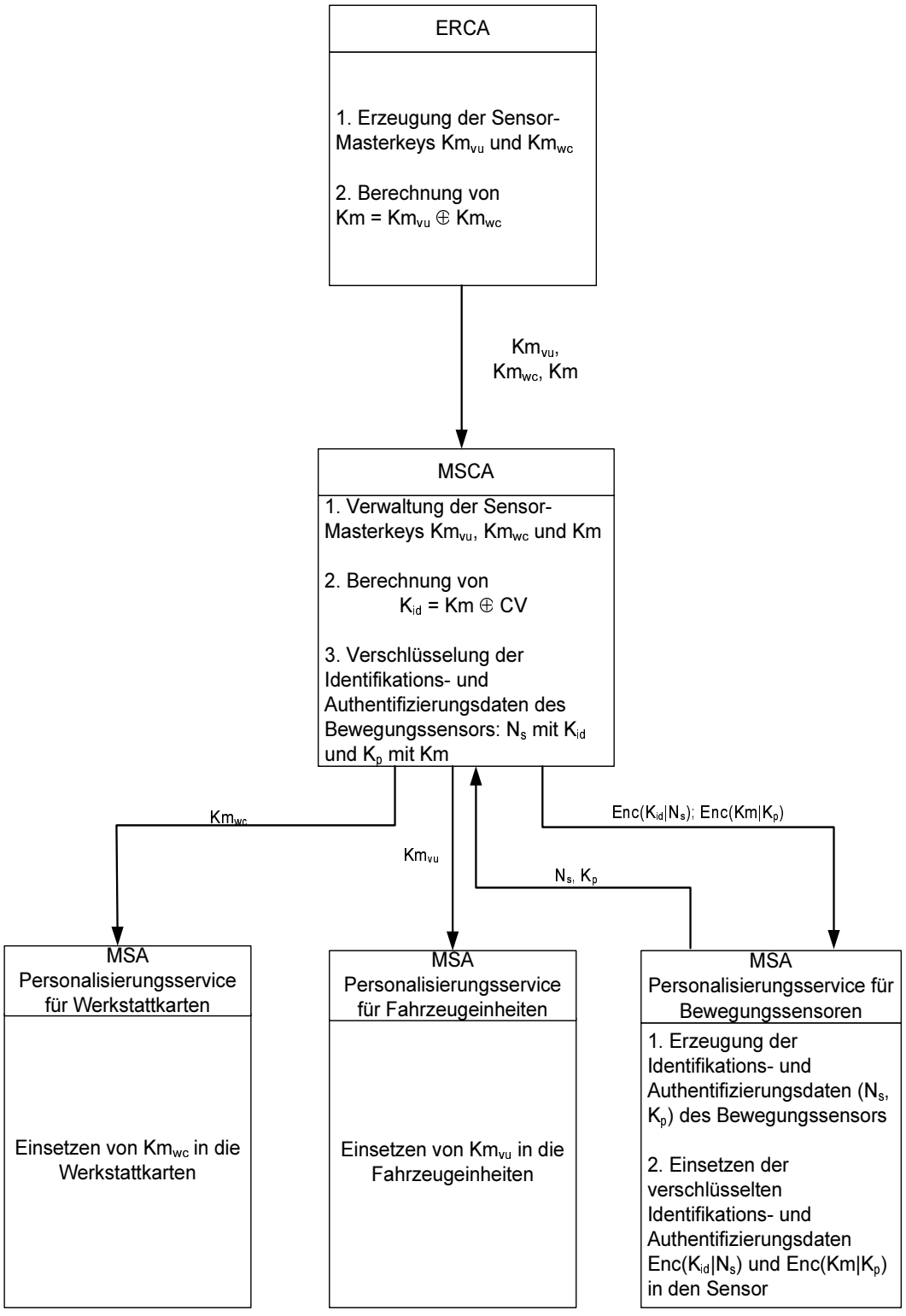

Abbildung 4.1: Schlüsselmanagement der Triple-DES-Schlüssel

Die MSCA verschlüsselt nun unter Verwendung von Triple-DES die erweiterte Seriennummer N_s des Sensors mit dem Identifikationsschlüssel K_{id} und den Pairingkey K_p mit dem Masterkey Km. Die Ergebnisse dieser Verschlüsselung, $Enc(K_{id}|N_s)$ und $Enc(Km|K_p)$, werden dann an die MSA zurück übermittelt und ebenfalls im Sensor gespeichert, sodaß er nach der Herstellung eine Seriennummer N_s, einen Pairingkey K_p sowie die mit K_{id} bzw. Km verschlüsselten Werte $Enc(K_{id}|N_s)$ und $Enc(Km|K_p)$ enthält.

Km und K_{id} werden ausschließlich während des *Pairings* (siehe Abschnitt 4.2) des Sensors mit einer Fahrzeugeinheit verwendet und weder im Sensor noch in der Fahrzeugeinheit gespeichert.

4.1 Allgemeiner Ablauf der Kommunikation

Bei der Kommunikation zwischen Fahrzeugeinheit (VU) und Bewegungssensor (MS) spielt letzterer immer eine passive Rolle, wohingegen die Fahrzeugeinheit die aktive Rolle übernimmt und Instruktionen an den Sensor schickt. Der allgemeine Ablauf der Kommunikation zwischen VU und MS ist schematisch in Abb. 4.4 dargestellt (siehe [Inte03]).

Die Datenübertragung zwischen Fahrzeugeinheit und Bewegungssensor erfolgt seriell und asynchron mit einer Übertragungsrate von 1200 Baud. Die übertragenen Datenrahmen haben die in Abb. 4.2 dargestellte Form. Sie bestehen aus acht Datenbits D_0 bis D_7, einem Startbit, einem Stop- und einem Paritätsbit.

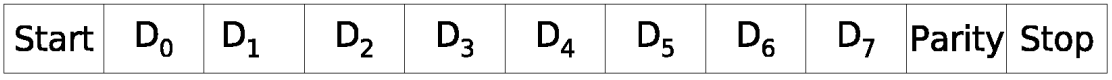

Abbildung 4.2: Struktur eines zwischen Fahrzeugeinheit und Bewegungssensor übertragenen Datenrahmens

Die Struktur einer gesamten Nachricht ist Abb. 4.3 zu entnehmen. Der Rahmen einer solchen Nachricht besteht aus einem sogenannten *Header*, der wiederum aus vier verschiedenen Feldern besteht. Das erste, *Sync*, ist ein Byte zur Synchronisation, das einen binären Wert bis zum dezimalen Wert 192 besitzt und der Kontrolle der Baud-Rate dient. Das nächste Feld, *Target*, ist ebenfalls ein Byte groß und dient der Identifikation der Übertragungsrichtung. Es enthält eine 0, falls die Richtung von der Fahrzeugeinheit zum Bewegungssensor ist, d.h. wenn der Sensor das Ziel ist, im anderen Fall eine 1. Das *STX*-Byte ist eine Konstante mit dem dezimalen Wert 2. Das Feld *Length* spezifiziert schließlich die Länge der gesamten Nachricht vom Sync- bis einschließlich LRC-Byte.

Im Anschluß an den Header kommen die Datenbytes, die die eigentliche Information enthalten, die übertragen werden soll. Sie haben die oben beschriebene Form. Schließlich enthält der Nachrichtenrahmen im Feld *Tail* noch ein *ETX*-Byte mit dem konstanten dezimalen Wert 3. Das *LRC*-Byte ist eine XOR-Verknüpfung aller Felder von Sync bis ETX.

Header				Data bytes				Tail	
Sync	Target	STX	Length	Instr. No.	Data	• • •	Data	ETX	LRC
LRC = XOR from Sync to ETX									

Abbildung 4.3: Struktur einer Nachricht

Ist die übertragene Nachricht ein *Request* von der Fahrzeugeinheit an den Sensor, so enthält die Nachricht zwischen *Length* und den eigentlichen Daten zusätzlich noch ein Feld mit der Nummer der entsprechenden Instruktion.

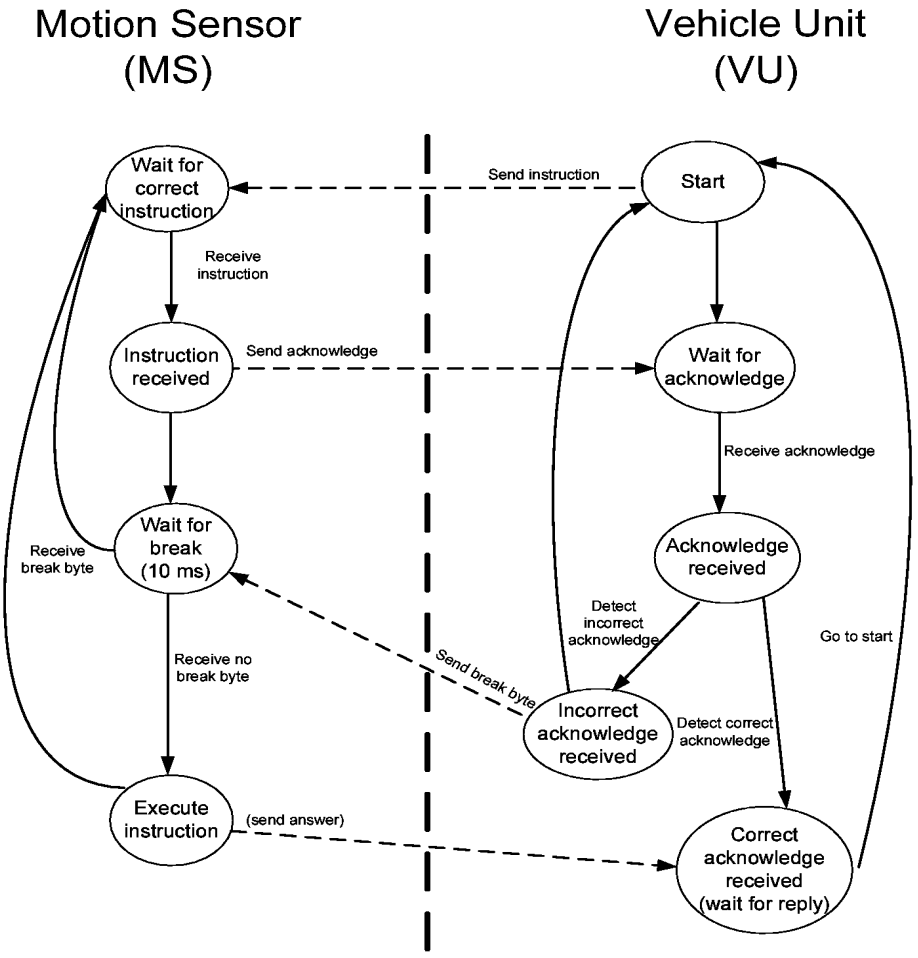

Abbildung 4.4: Ablauf der Kommunikation zwischen Sensor und Fahrzeugeinheit

4.2 Pairing

Die Kommunikation zwischen der Fahrzeugeinheit und dem Sensor besteht im Wesentlichen aus zwei Phasen[1] [LePW06]: dem sogenannten *Pairing* (dt. Paarung), bei dem ein gemeinsamer Triple-DES-Sitzungsschlüssel K_s vereinbart wird, und der *Betriebsphase*, in der dieser Sitzungsschlüssel zur weiteren sicheren Kommunikation verwendet wird.

[1] Gemäß ISO 16844-3 „Motion Sensor Interface“

Bevor das Pairing zwischen der Fahrzeugeinheit (VU) und dem Sensor (MS) stattfinden kann, muß zunächst eine gültige und authentische Werkstattkarte in die VU eingesteckt werden. Nach einer wechselseitigen Authentifizierung zwischen der Werkstattkarte und der Fahrzeugeinheit, liest die Fahrzeugeinheit den Teilschlüssel Km_{wc} aus der Werkstattkarte und berechnet daraus den *Masterkey* Km als

$$Km = Km_{vu} \oplus Km_{wc},$$

sowie den *Identifikationsschlüssel* K_{id} als XOR-Verknüpfung des Masterkeys mit einem konstanten Kontrollvektor CV, d.h.

$$K_{id} = Km \oplus CV,$$

wobei $\oplus$ die XOR-Verknüpfung ist. Der Kontrollvektor CV[2] ist ebenfalls in der Fahrzeugeinheit gespeichert. Km und K_{id} werden anschließend weder in der Fahrzeugeinheit noch im Bewegungssensor gespeichert. Sobald die Werkstattkarte wieder aus der Fahrzeugeinheit entnommen wird, gehen sie verloren.

4.2.1 Authentifizierung der Fahrzeugeinheit

Nachdem die Fahrzeugeinheit mit Hilfe des in der Werkstattkarte gespeicherten Teilschlüssels Km_{wc} den Masterkey Km und den Identifikationsschlüssel K_{id} berechnet hat, erfolgt als erster Schritt der Kommunikation zwischen Fahrzeugeinheit (VU) und Bewegungssensor (MS) zunächst die Authentifizierung der VU gegenüber dem Sensor durch ein *Challenge-Response*-Verfahren, das durch Senden einer speziellen Instruktion von der VU an den Sensor initialisiert wird. Der Sensor schickt daraufhin seine intern gespeicherte erweiterte Seriennummer N_s als „Herausforderung“ (engl. challenge) im Klartext an die VU. Diese verschlüsselt N_s mit K_{id} und sendet das Ergebnis $Enc(K_{id}|N_s)$ als Antwort (engl. response) an den Sensor zurück. Der vergleicht den empfangenen Wert mit seinem intern gespeicherten Wert $Enc(K_{id}|N_s)$. Bei Übereinstimmung wird die Authentifizierung der VU als korrekt angenommen und der erste Teil der Kommunikation ist abgeschlossen.

4.2.2 Etablierung des Sitzungsschlüssels

Nach der erfolgreichen Authentifizierung der Fahrzeugeinheit gegenüber dem Sensor erfolgt im nächsten Schritt die Erzeugung eines gemeinsamen *Sitzungsschlüssels* (engl. session key) K_s. Dazu sendet der Sensor seinen intern gespeicherten Wert $Enc(Km|K_p)$ an die VU. Dies ist der mit dem Masterkey Km verschlüsselte, eindeutige *Pairingkey* K_p des Sensors. Die VU entschlüsselt K_p mit Hilfe von Km und generiert anschließend einen zufälligen Triple-DES-Sitzungsschlüssel K_s für die Betriebsphase. Diesen Sitzungsschlüssel K_s sendet die VU mit K_p verschlüsselt an den Sensor. Dieser entschlüsselt K_s und speichert ihn permanent in seinem nichtflüchtigen Speicher. Damit ist die Vereinbarung eines Sitzungsschlüssels abgeschlossen. Er wird später in der Betriebsphase für alle nachfolgenden kryptographischen Operationen benutzt.

[2] besitzt den hexadezimalen Wert 48 21 5F 00 03 41 32 8A 00 68 4D 00 CB 21 70 1D

4.2.3 Austausch der Pairinginformation und Authentifizierung des Sensors

Nach der Erzeugung eines gemeinsamen Sitzungsschlüssels K_s erfolgt nun das Pairing zwischen Fahrzeugeinheit und Sensor. Dazu verschlüsselt die VU die sogenannte *Pairinginformation* (engl. pairing data) P_D mit Triple-DES unter Verwendung des Pairingkeys K_p und sendet das Ergebnis $Enc(K_p|P_D)$ an den Sensor. Die Pairinginformation besitzt eine Länge von 24 Byte und besteht aus dem Datum des Pairings, der Bauartzulassungsnummer (engl. type approval number) sowie der Seriennummer der Fahrzeugeinheit. Der Sensor entschlüsselt die erhaltene Pairinginformation mit Hilfe des Pairingkeys K_p und speichert sie in seinem nichtflüchtigen Speicher.

Anschließend fordert die Fahrzeugeinheit die Authentifizierung des Sensors an. Dieser antwortet, indem er die Pairinginformation P_D mit dem Sitzungsschlüssel K_s verschlüsselt und an die VU sendet. Die Fahrzeugeinheit entschlüsselt diese und vergleicht den so erhaltenen Wert mit der aktuellen Pairinginformation. Bei Übereinstimmung wird angenommen, daß der Sensor den korrekten Sitzungsschlüssel verwendet und die Authentifizierung korrekt ist. Nach der erfolgreichen gegenseitigen Authentifizierung, der Erzeugung eines gemeinensamen Sitzungsschlüssels und dem Pairing, wird die Kommunikation zwischen der Fahrzeugeinheit und dem Sensor in der Betriebsphase mit dem Sitzungsschlüssel K_s fortgesetzt.

4.3 Betrieb

In der Betriebsphase werden unter Verwendung des Sitzungsschlüssels K_s zwischen der Fahrzeugeinheit und dem Bewegungssensor drei verschiedene Arten von Daten übermittelt:

- Echtzeit-Bewegungsimpulse
- verschlüsselter Wert des Impulszählers
- verschlüsselter Inhalt der Sensordateien

Während der Bewegung des Fahrzeugs, werden kontinuierlich Bewegungsimpulse in Echtzeit und ohne jegliche Sicherheitsmechanismen vom Sensor an die Fahrzeugeinheit übermittelt. Die Frequenz der Impulse ist dabei abhängig von der momentanen Geschwindigkeit des Fahrzeugs und der Art und Weise, wie der Sensor im Getriebe montiert ist. Die für die korrekte Umrechnung benötigten Koeffizienten werden während der Kalibrierung von einer geprüften Werkstatt in der Fahrzeugeinheit gespeichert.

Sowohl der Sensor als auch die Fahrzeugeinheit besitzen jeweils einen Impulszähler, mit dem die Anzahl der in einem bestimmten Zeitraum gesendeten bzw. empfangenen Impulse gemessen wird. Die Werte dieser beiden Zähler werden unmittelbar nach dem Pairing synchronisiert. Die Fahrzeugeinheit sendet dann in regelmäßigen Abständen - mindestens einmal pro Stunde - eine Authentifizierungsaufforderung an den Sensor. Diese Anforderung (engl. request) besteht aus einer 4 Byte langen Zufallszahl und zusätzlich 4 Byte Kontrollinformation. Der Sensor fängt daraufhin den aktuellen Zählerstand ein und verschlüsselt diesen mit dem Sitzungsschlüssel

K_s. Außerdem berechnet er die XOR-Verknüpfung der erhaltenen Zufallszahl mit seiner Seriennummer N_s und verschlüsselt dies ebenfalls mit K_s. Beide Werte sendet er dann an die Fahrzeugeinheit zurück. Die Fahrzeugeinheit vergleicht die erhaltenen Werte mit der von ihr gesendeten Zufallszahl und dem Wert ihres eigenen Impulszählers. Bei Übereinstimmung der Werte kann die Fahrzeugeinheit davon ausgehen, daß der angeschlossene Sensor korrekt arbeitet und keine Impulse eingefügt oder ausgelassen worden sind. Auf diese Weise stellt der digitale Tachograph die Korrektheit des Mittelwerts der Bewegungsdaten zwischen zwei aufeinanderfolgenden Anforderungen des verschlüsselten Wertes des Impulszählers sicher. Ein Teil der Daten, die permanent im Bewegungssensor gespeichert sind wie z.B. Fehlermeldungen, Seriennummer und Pairinginformationen sind in Dateien organisiert, die von der angeschlossenen Fahrzeugeinheit ausgelesen werden können. Auf eine spezielle Anforderung hin sendet der Sensor den Inhalt dieser Dateien verschlüsselt mit dem Sitzungsschlüssel K_s an die Fahrzeugeinheit.

4.4 Prägung der Fahrzeugeinheit

Um zu verhindern, daß der Sensor, der die Impulse zur Weg- und Geschwindigkeitsmessung an die Fahrzeugeinheit sendet, gegen einen manipulierten Sensor, der weniger Impulse sendet, ausgetauscht wird, ist ein Verfahren nötig, das die Fahrzeugeinheit so mit einem bestimmten Sensor verbindet, daß ein Austauschen dieses Sensors erkannt und somit unmöglich gemacht wird.

Eine Möglichkeit dies zu erreichen, ist die von Ross Anderson in [Ande01] beschriebene Methode des *Resurrecting Duckling*. Ähnlich einem Küken, das nach dem Schlüpfen durch einen Laut auf das erste sich bewegende Objekt, das es sieht, geprägt wird und dieses als seine Mutter annimmt, wird hier der erste Sensor, der nach dem Einschalten einer „wiedergeborenen“, d.h. aus der Verpackung entnommenen Fahrzeugeinheit, einen geheimen Schlüssel an diese sendet, von ihr als „Mutter“ (Master) anerkannt. Der Sensor tut dies beim Einschalten automatisch. Bei der Prägung wird ein sogenannter *Prägeschlüssel* ausgetauscht und die Fahrzeugeinheit bleibt für den Rest ihres „Lebens“ bzw. bis zu einer Reinitialisierung auf diesen Sensor geprägt und vertraut ihm.

Falls ein Sensor ausfällt und ausgetauscht werden muß, wird die Prägung der Fahrzeugeinheit gelöscht. Sie wird dadurch zurückgesetzt und kann dann als „neugeborene“ (resurrecting) Fahrzeugeinheit auf einen anderen Sensor geprägt werden. Diese „Neuprägung“ kann jedoch nur von autorisierten Personen in einer Werkstatt mit einer gültigen Werkstattkarte durchgeführt werden. Dadurch wird verhindert, daß ein Angreifer die Fahrzeugeinheit zurücksetzen und auf einen anderen (manipulierten) Sensor prägen kann. Die Prägung muß daher so stark sein, daß es sich für einen Angreifer nicht lohnt, das System zu übernehmen, da dies zur Zerstörung des Systems führen würde.

5. Der Data Encryption Standard (DES)

5.1 Geschichte

Der *Data Encryption Standard* (DES) ging als Sieger einer Ausschreibung des *National Bureau of Standards (NBS)*, heute *National Institute of Standards and Technology (NIST)* für ein sicheres kryptographisches Verfahren zur Speicherung und Übermittlung von Daten hervor. Das NBS gab damit den Anstoß zur Entwicklung eines einheitlichen und standardisierten kryptographischen Algorithmus. Die Sicherheit sollte dabei ausschließlich auf der Geheimhaltung des verwendeten Schlüssels und nicht des Algorithmus basieren *(Kerckhoffs-Prinzip)*.

Da keiner nach der öffentlichen Ausschreibung im Jahre 1973 eingereichten Vorschläge die Voraussetzungen erfüllte, gab es im darauffolgenden Jahr eine zweite Ausschreibung, woraufhin von IBM ein vielversprechender Vorschlag eingereicht wurde, der auf dem kurz zuvor von Horst Feistel entwickelten Algorithmus *Lucifer* basierte.

Unter dem Einfluss der *National Security Agency (NSA)*, die den Sicherheitsinteressen der USA dient, wurden einige bis heute umstrittene Änderungen vorgenommen (unter anderem wurde die Schlüssellänge von 128 auf 64 Bit verkürzt) und der Algorithmus 1977 zum Data Encryption Standard erklärt und veröffentlicht. Er war von da an einer der wichtigsten und am häufigsten eingesetzten kryptographischen Algorithmen für nichtmilitärische Anwendungen weltweit.

Erst als es der *Elecronic Frontier Foundation (EFF)* 1998 gelang, durch einen *Brute-Force-Angriff* einen Schlüssel zu knacken, galt DES als nicht mehr sicher genug und es wurde nach einem Nachfolger gesucht, der im Jahr 2000 als *Advanced Encryption Standard (AES)* verabschiedet wurde.

Die Weiterentwicklung von DES, *Triple-DES*, wird dagegen bis heute eingesetzt, da sie aufgrund der dreimaligen Hintereinanderausführung von DES und der doppelten oder dreifachen Schlüssellänge immernoch ausreichenden Schutz bietet und derzeit noch nicht mit einem Brute-Force-Angriff geknackt werden kann. Eine genaue Beschreibung von DES findet sich unter anderem in [Nati99] und [Schn93].

5.2 Funktionsweise und Ablauf der Verschlüsselung

DES ist eine symmetrische, iterierte Blockchiffre und arbeitet mit Blöcken der Länge 64 Bit und einer effektiven Schlüssellänge von 56 Bit. Seine Struktur ist schematisch in Abb. 5.1 dargestellt.

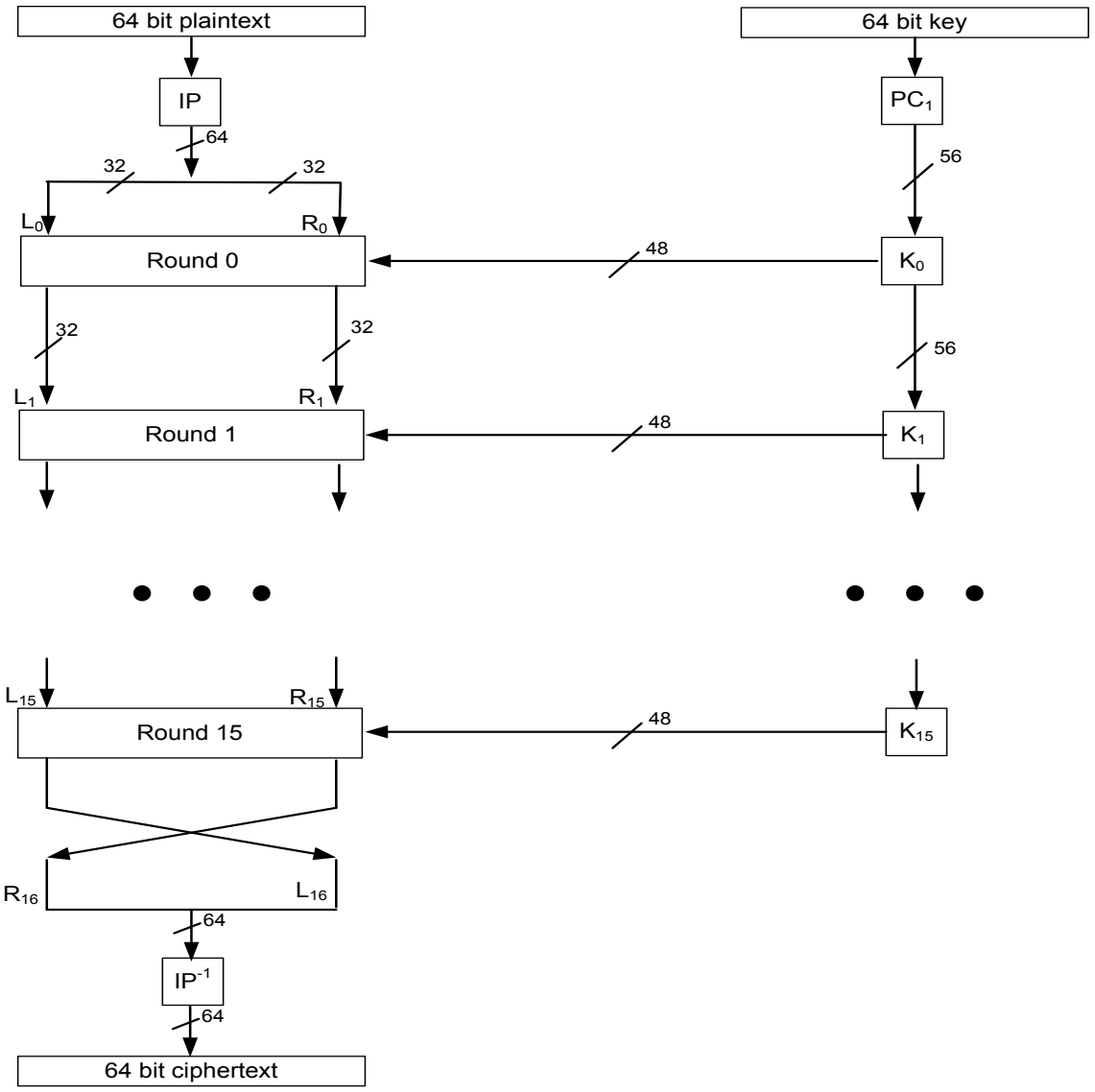

Abbildung 5.1: Ablauf des DES-Algorithmus

Auf den *Klartext* (engl. plaintext) $p = (p_1, ..., p_{64})$ wird zunächst die *Eingangspermutation* (engl. initial permutation) IP angewendet und man erhält den *permutierten Klartext* (engl. permuted plaintext) $p_p = (p_{p1}, ..., p_{p64}) = \mathrm{IP}(p)$. Dieser wird dann in zwei gleichgroße Hälften $L_0 = (p_{p1}, ..., p_{p32})$ und $R_0 = (p_{p33}, ..., p_{p64})$ zerlegt, sodaß gilt: $\mathrm{IP}(p) = (L_0||R_0)$[1].

Anschließend werden 16 identische Runden durchlaufen, wobei die Eingabeblöcke nach dem Prinzip von Feistel verarbeitet und mit verschiedenen Teilen des Schlüssels, den sogenannten *Rundenschlüsseln* K_i, verschlüsselt werden. Nach den 16 Runden

[1] wobei || die Konkatenation ist

werden die beiden Hälften vertauscht, konkateniert und der *Ausgangspermutation* (engl. final permutation) IP^{-1} unterzogen, die invers zu IP ist. Das *Chiffrat* (engl. ciphertext) $c = (c_1, ..., c_{64})$ berechnet sich somit zu

$$c = \mathrm{IP}^{-1}(R_{16}||L_{16}).$$

Innerhalb der i-ten Runde werden dabei für die linke und rechte Hälfte folgende Formeln verwendet (siehe Abb. 5.2):

$$L_i = R_{i-1}$$
$$R_i = L_{i-1} \oplus f(R_{i-1}, K_i)$$

Dabei ist K_i der i-te Rundenschlüssel und f die sogenannte *Rundenfunktion*.

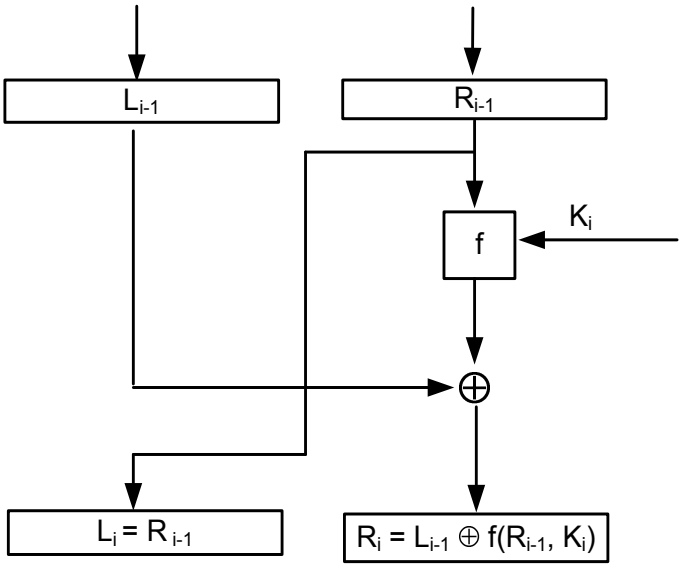

Abbildung 5.2: Struktur einer Runde von DES

Die Rundenfunktion f besteht aus verschiedenen, hintereinander ausgeführten Operationen, die schematisch in Abb. 5.3 dargestellt sind: die erste ist eine *Expansion* der 32 Bit der rechten Seite R_i auf 48 Bit. Anschließend werden diese 48 Bit mit dem ebenfalls 48 Bit langen Rundenschlüssel K_i durch XOR verknüpft und dann in acht sogenannten *S-Boxen* (Substitutionsboxen) wieder auf 32 Bit abgebildet. Jede S-Box erhält dabei 6 Bit als Eingabe und liefert 4 Bit als Ausgabe. Schließlich erfolgt noch eine Permutation, die *P-Permutation*, in der die 32 Bits nochmals vermischt werden.

Insgesamt ist die Rundenfunktion f des DES somit definiert als

$$f(R_{i-1}, K_i) = P(S(E(R_{i-1}) \oplus K_i)),$$

wobei E die Expansion, S die Substitution durch die S-Boxen und P die Permutation nach den S-Boxen ist.

Die jeweiligen Rundenschlüssel K_i werden aus dem Schlüssel $k = (k_1, ..., k_{64})$ wie folgt generiert: der 64 Bit lange Schlüssel wird zunächst durch die Schlüsselpermutation PC_1 permutiert und auf 56 Bit verkürzt, indem jedes achte Bit ignoriert wird, da es als Paritätsbit dient. Man erhält so den permutierten Schlüssel

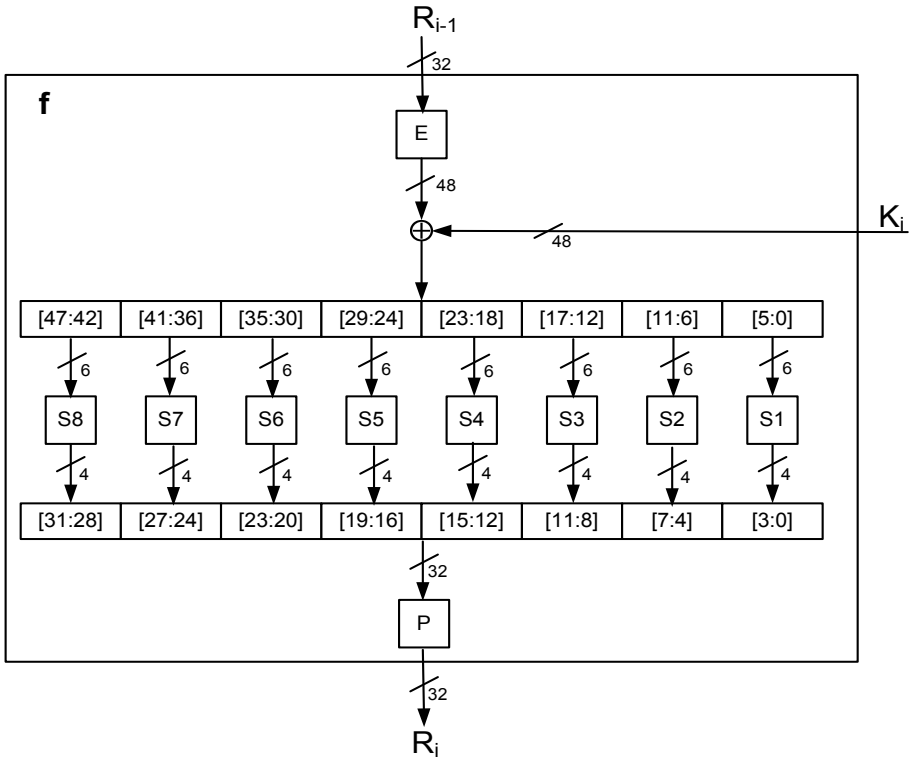

Abbildung 5.3: Die Rundenfunktion f des DES-Algorithmus

$PC_1(k) = k_p = (k_{p1}, ..., k_{p28}, k_{p29}, ..., k_{p56})$, der ebenfalls in zwei Hälften C_0 und D_0 zerlegt wird, sodaß gilt $C_0 = (k_{p1}, ..., k_{p28})$ und $D_0 = (k_{p29}, ..., k_{p56})$.

Die Hälften werden nun in jeder Runde abhängig vom Rundenindex jeweils um ein oder zwei Bit zyklisch nach links verschoben (engl. cyclic shift left, CSL). Abb. 5.4 zeigt eine solche Verschiebung um ein Bit. Dabei enthält das Register C die niederwertigen 28 Bit b_1 bis b_{28} und D die höherwertigen b_{29} bis b_{56} des permutierten Schlüssels. Oben ist jeweils die Position der Bits b_i in diesem Register angegeben.

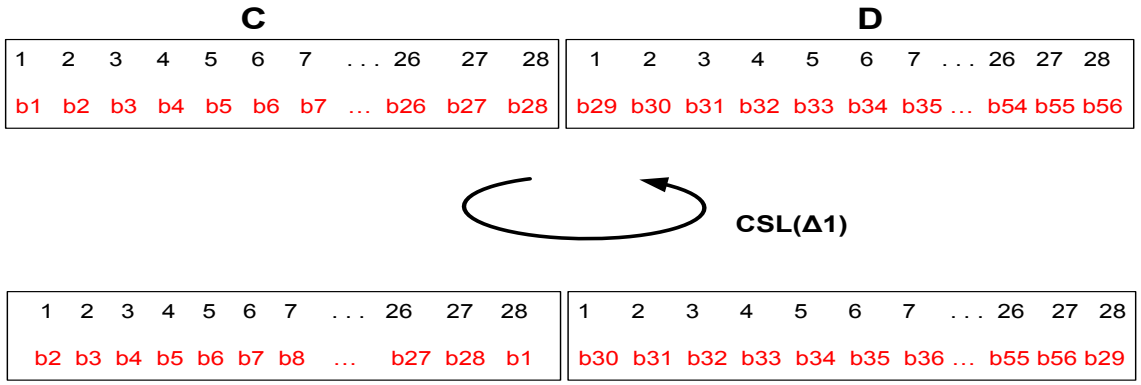

Abbildung 5.4: Zyklische Verschiebung der Schlüsselbits um ein Bit nach links

Die Anzahl $\triangle n$ der Bits, um die der Schlüssel pro Runde verschoben wird, ist der folgenden Tabelle zu entnehmen. Dabei ist n der Index der jeweiligen Runde.

Runde n	$\triangle n$
1	1
2	1
3	2
4	2
5	2
6	2
7	2
8	2
9	1
10	2
11	2
12	2
13	2
14	2
15	2
16	1

Nach der zyklischen Verschiebung der Schlüsselbits werden durch die Funktion PC_2 48 der 56 Bit als Rundenschlüssel ausgewählt. PC_2 ist ebenso wie PC_1 eine *permutierte Auswahl* (engl. permuted choice), da sie sowohl die Reihenfolge der Bits ändert als auch eine Teilmenge auswählt. Sie kann geschrieben werden als $PC_2(C_i||D_i) = K_i$, wobei K_i der Rundenschlüssel der i-ten Runde ist.

5.3 Entschlüsselung einer Feistel-Struktur

Der Data Encryption Standard basiert auf einer Feistel-Struktur, wie sie in Abb. 5.2 dargestellt ist. Diese Chiffren besitzen die Eigenschaft, daß zur Entschlüsselung derselbe Algorithmus und dieselbe Rundenfunktion f verwendet werden wie bei der Verschlüsselung. Die Rundenfunktion muß daher bei einer solchen Chiffre nicht umkehrbar sein. Dadurch müssen Ver- und Entschlüsselung auch nicht getrennt implementiert werden. Es werden lediglich die Rundenschlüssel in umgekehrter Reihenfolge angewendet.

Bei der Generierung der Rundenschlüssel für die Entschlüsselung werden die Schlüsselbits daher nach rechts und nicht nach links verschoben. Die zyklische Verschiebung um ein Bit nach rechts (engl. cyclic shift right, CSR) ist in Abb. 5.5 dargestellt.

Die folgende Tabelle gibt die Anzahl $\triangle n$ der Bits an, um die der Schlüssel bei der Entschlüsselung pro Runde verschoben wird:

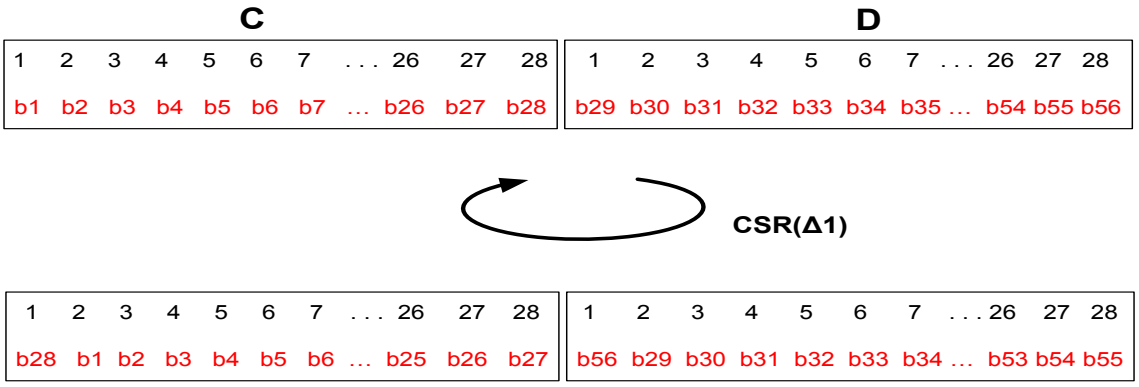

Abbildung 5.5: Zyklische Verschiebung der Schlüsselbits um ein Bit nach rechts

Runde n	$\triangle n$
1	0
2	1
3	2
4	2
5	2
6	2
7	2
8	2
9	1
10	2
11	2
12	2
13	2
14	2
15	2
16	1

Bei einer Verschlüsselung wurden die Schlüsselbits nach 16 Runden um insgesamt 28 Bit verschoben (vgl. Tabelle), da dort gilt $\triangle 1 + \triangle 2 + ... + \triangle 16 = 28$. Sie befinden sich damit also wieder an derselben Position, wie zu Beginn, d.h. nach der Permutation PC_1. Der 16. Rundenschlüssel K_{16} ist im Fall der Verschlüsselung also gleich $PC_1(k)$, wobei k der verwendete Schlüssel ist. Dementsprechend wurden die Schlüsselbits bis zur 15. Runde um insgesamt 27 Bit nach links verschoben, da $\triangle 16 = 1$ und $\triangle 1 + \triangle 2 + ... + \triangle 15 = 27$..

Da die beiden Hälften C und D nur jeweils 28 Bit groß sind (vgl. Abb. 5.4), entspricht dies gleichzeitig einer Verschiebung um ein Bit nach rechts. Eine Verschiebung um insgesamt 25 Bit nach links (bis zur 14. Runde) entspricht einer Verschiebung um 3 Bit nach rechts usw.

Bei der Entschlüsselung werden die Rundenschlüssel also in umgekehrter Reihenfolge angewendet, sie erfolgt damit im Vergleich zur Verschlüsselung quasi rückwärts: In der ersten Runde wird der Schlüssel - wie in obiger Tabelle ersichtlich - gar nicht verschoben. Er ist also gleich $PC_1(k)$ und damit gleich dem 16. Rundenschlüssel K_{16} bei der Verschlüsselung. Der zweite Rundenschlüssel wurde um ein Bit nach rechts verschoben, was einer Verschiebung um 27 Bit nach links und damit dem

15. Rundenschlüssel der Verschlüsselung entspricht usw. Nach 16 Runden wurden die Schlüsselbits um insgesamt 27 Bit nach rechts geschoben, was wiederum einer Verschiebung um ein Bit nach links und damit dem ersten Rundenschlüssel bei der Verschlüsselung entspricht.

Zur Entschlüsselung werden also derselbe Algorithmus und dieselbe Rundenfunktion verwendet. Lediglich die Anwendung der Rundenschlüssel erfolgt in umgekehrter Reihenfolge. Dies wird durch die Eigenschaft einer Feistel-Chiffre gewährleistet.

5.4 Triple-DES

Im Jahr 1998 wurde von der Electronic Frontier Foundation (EFF) durch einen Brute-Force-Angriff, d.h. durch Ausprobieren aller 2^{56} möglichen Kombinationen, ein Schlüssel von DES geknackt. Schon bei seiner Veröffentlichung war dessen effektive Schlüssellänge von nur 56 Bit als zu kurz kritisiert worden, da die Sicherheit des Algorithmus ausschließlich auf der Geheimhaltung des Schlüssels basiert. DES war daher nur so lange sicher bis es genug Rechenleistung gab um alle möglichen Schlüssel in adäquater Zeit auszuprobieren. Nach dem erfolgreichen Angriff der EFF galt DES als nicht mehr sicher genug und es wurde nach einem Nachfolger gesucht. Bis dieser gefunden war wurde eine Weiterentwicklung von DES, genannt *Triple-DES*, eingesetzt, bei der DES dreimal hintereinander unter Verwendung von mindestens zwei verschiedenen Schlüsseln ausgeführt wird, sodaß die effektive Schlüssellänge mindestens 112 Bit beträgt.

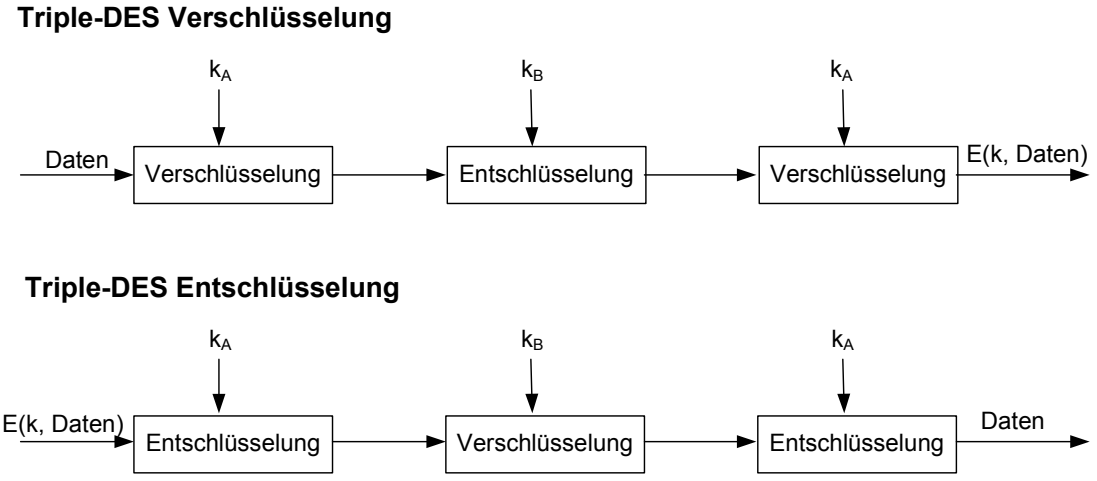

Abbildung 5.6: Ablauf der Ver- und Entschlüsselung des Triple-DES

Der Ablauf der Variante *2TDES* von Triple-DES mit zwei verschiedenen Schlüsseln k_A und k_B ist in Abb. 5.6 dargestellt. Dabei werden die eingegebenen Daten zunächst mit k_A verschlüsselt, das Ergebnis dieser Verschlüsselung danach mit k_B entschlüsselt und anschließend noch einmal mit k_A verschlüsselt. Der Ablauf der Entschlüsselung ist genau umgekehrt: es wird zuerst mit k_A entschlüsselt, dann mit k_B ver- und anschließend noch einmal mit k_A entschlüsselt.

Im Jahr 2000 wurde der *Advanced Encryption Standard (AES)* als offizieller Nachfolger von DES verabschiedet und veröffentlicht. Triple-DES wird jedoch weiterhin eingesetzt, da die Schlüssellänge von mindestens 112 Bit auch in den nächsten Jahren noch nicht durch einen Brute-Force-Angriff zu knacken sein wird.

6. Seitenkanalangriffe und Gegenmaßnahmen

6.1 Einführung in die Kryptoanalyse

Die *Kryptographie* ist die Wissenschaft von der Verschlüsselung von Informationen. Sie dient dem Schutz von Daten durch deren Transformation, in der Regel unter Einbeziehung von geheimen Schlüsseln. Weitere Einsatzgebiete der Kryptographie sind[1]:

- *Vertraulichkeit:* Nur berechtigte Personen sollen in der Lage sein, die Daten oder die Nachricht zu lesen oder Informationen über deren Inhalt zu erlangen.
- *Integrität:* Der Empfänger sollte in der Lage sein festzustellen, ob die Daten oder die Nachricht nach ihrer Erzeugung verändert wurden.
- *Authentizität:* Der Urheber der Daten bzw. der Absender der Nachricht sollte eindeutig identifizierbar und seine Urheberschaft nachprüfbar sein.
- *Verbindlichkeit:* Der Urheber von Daten oder der Absender einer Nachricht sollte nicht in der Lage sein, seine Urheberschaft zu bestreiten, d.h. sie sollte sich gegenüber Dritten nachweisen lassen.

Kryptographische Verfahren dienen dabei nicht notwendigerweise allen genannten Zielen.

Das „Gegenstück“ der Kryptographie ist die *Kryptoanalyse*. Sie bezeichnet die Studie von Methoden und Techniken zur Gewinnung von Informationen aus verschlüsselten Texten ohne Kenntnis des geheimen Schlüssels. Beide sind Teilgebiete der *Kryptologie*. Heutzutage bezeichnet der Begriff Kryptoanalyse allgemeiner die Analyse von kryptographischen Verfahren (nicht nur zur Verschlüsselung) mit dem Ziel, diese entweder zu „brechen“, d.h. ihre Schutzfunktion aufzuheben bzw. zu umgehen, oder ihre Sicherheit nachzuweisen und zu quantifizieren[2].

[1] http://de.wikipedia.org/wiki/Kryptographie
[2] http://de.wikipedia.org/wiki/Kryptoanalyse

Je nachdem auf welche Implementierungsform eines kryptographischen Verfahrens ein Angriff abzielt, werden unterschiedliche Methoden zur Kryptoanalyse eingesetzt.

6.2 Implementierungsformen kryptographischer Verfahren

Für die Implementierung kryptographischer Verfahren gibt es verschiedene Möglichkeiten. Abb. 6.1 gibt einen Überblick über die verschiedenen Implementierungsformen.

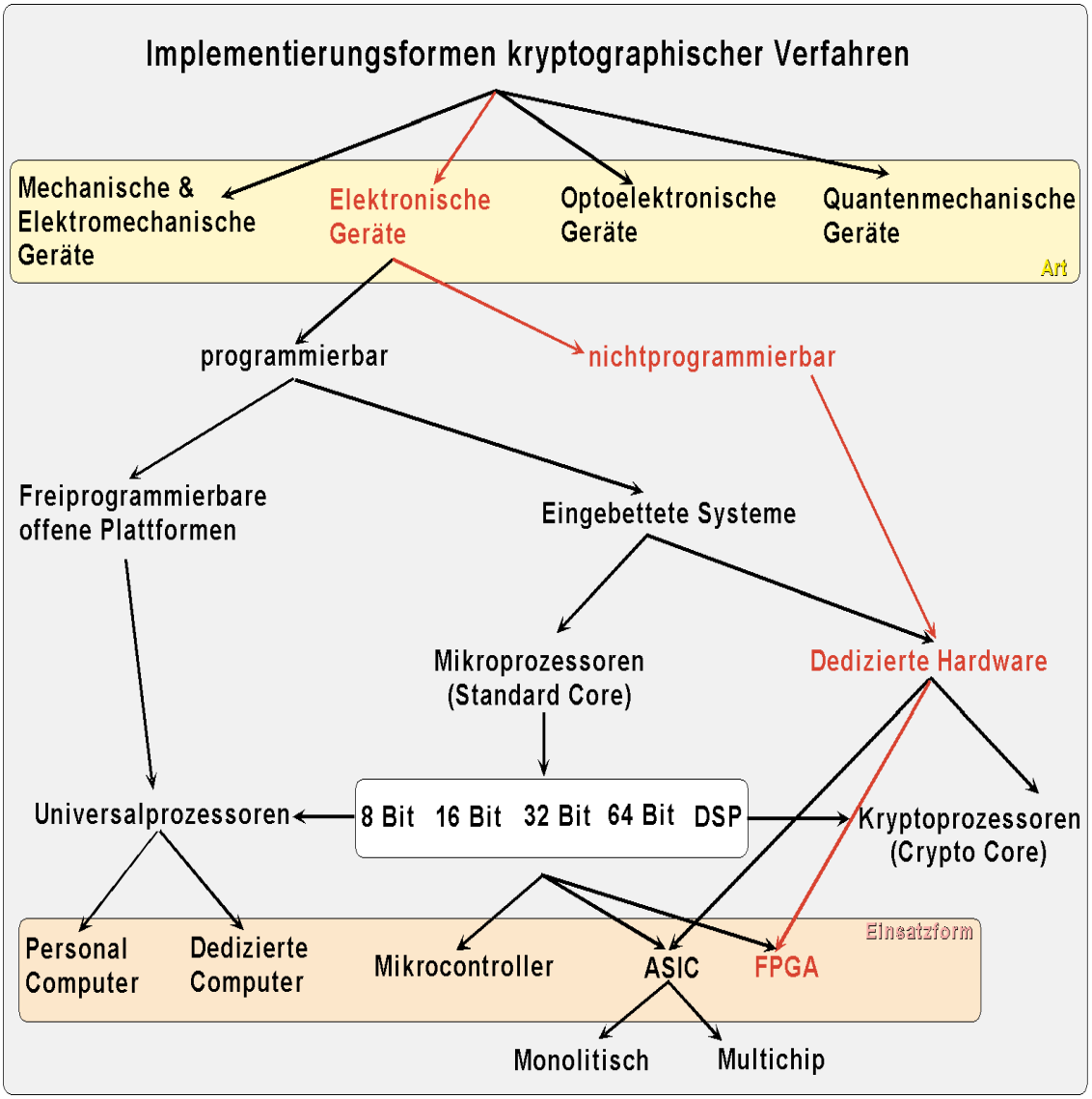

Abbildung 6.1: Implementierungsformen kryptographischer Verfahren

Zu den mechanischen und elektromechanischen Geräten zählt z.B. die im zweiten Weltkrieg zur Verschlüsselung eingesetzte Maschine *Enigma*.

Heute werden Implementierungen kryptographischer Verfahren vorwiegend in elektronischen Geräten auf offenen Plattformen oder in eingebetteten Systemen vorgenommen. Offene Plattformen sind dabei PCs oder Workstations, die einen Universalprozessor besitzen und viele verschiedene Anwendungen ausführen können. Moderne

Prozessoren (engl. central processing unit, CPU) sind universelle integrierte Schaltungen, die eine Vielzahl verschiedener Aufgaben bewältigen können.

Die PCs sind meist mit dem Internet verbunden, wodurch sie anfällig sind für Angriffe mit Viren oder Würmern, die über das Internet verbreitet werden. Die Systeme müssen daher zusätzlich vor solchen Gefahren geschützt werden. Sie haben dafür den Vorteil, daß viele verschiedene Anwendungen auf ihnen laufen und sie für unterschiedliche Zwecke genutzt werden können. Kryptographische Verfahren laufen hier meist als Programme in Software, die vom Prozessor ausgeführt werden. Der Prozessor ist dabei nicht für eine bestimmte Anwendung konzipiert, sondern kann verschiedene Arten von Programmen ausführen.

Im Gegensatz dazu wird ein *eingebettetes System* (engl. embedded system) nur für eine bestimmte Anwendung entworfen und in eine Umgebung oder ein größeres System eingebettet. Es wurde für die spezielle Anwendung optimiert und seine Funktion wird im Allgemeinen nachträglich nicht mehr geändert. Die Implementierung einer kryptographischen Funktion in einem solchen System erfolgt entweder in hardwarenaher Software als Mikroprogramm, das von einem *Mikroprozessor* ausgeführt wird, oder direkt in Hardware als (kombinatorische) digitale Schaltung oder endlicher Automat (engl. finite state machine).

Sowohl digitale Schaltungen als auch Mikroprozessoren können entweder in *anwendungsspezifischen integrierten Schaltkreisen* (engl. application specific integrated circuits, ASIC) oder in freiprogrammierbaren Logikschaltkreisen, sogenannten *FPGAs* (engl. field programmable gate array) realisiert werden. Ein ASIC wird dabei für eine bestimmte Anwendung entworfen und optimiert. Er ist fest verdrahtet, wodurch seine Funktionalität fest vorgegeben ist und nachträglich nicht mehr geändert werden kann. Eine solche Realisierung lohnt sich jedoch erst bei einer ausreichend hohen Stückzahl, da der Entwurf und das Design der Schaltkreise speziell für diese Anwendung vorgenommen werden müssen.

Im Gegensatz dazu ist ein FPGA reprogrammierbar und dadurch besonders zu Testzwecken und für die Evaluierung verschiedener Lösungen geeignet. Daher wurde für diese Arbeit auch die Realisierung auf einem FPGA gewählt.

6.3 Angriffe auf kryptographische Systeme

Je nach Implementierungsart und Plattform eines kryptographischen Systems gibt es verschiedene Möglichkeiten eines Angriffs. Abb. 6.2 gibt einen Überblick über die verschiedenen Arten von Angriffen.

Zu den Methoden der klassischen, theoretischen Kryptoanalyse gehört das analytische Vorgehen, bei dem die mathematischen oder theoretischen Schwachstellen eines kryptographischen Verfahrens oder Algorithmus untersucht werden. Eine weitere Möglichkeit, den geheimen Schlüssel eines Verfahrens zu enthüllen und es dadurch zu knacken ist das Durchprobieren aller möglichen Schlüssel (*Brute-Force*). Bei einer Schlüssellänge von k Bit sind dies 2^k verschiedene Schlüssel. In Zeiten wachsender Rechenleistung[3] stellt dies eine ernstzunehmende Bedrohung kryptographischer Verfahren dar, die nur durch eine ausreichend große Schlüssellänge abgewehrt werden kann.

[3] nach *Moore's Gesetz* verdoppelt sich die Rechenleistung alle 18 Monate

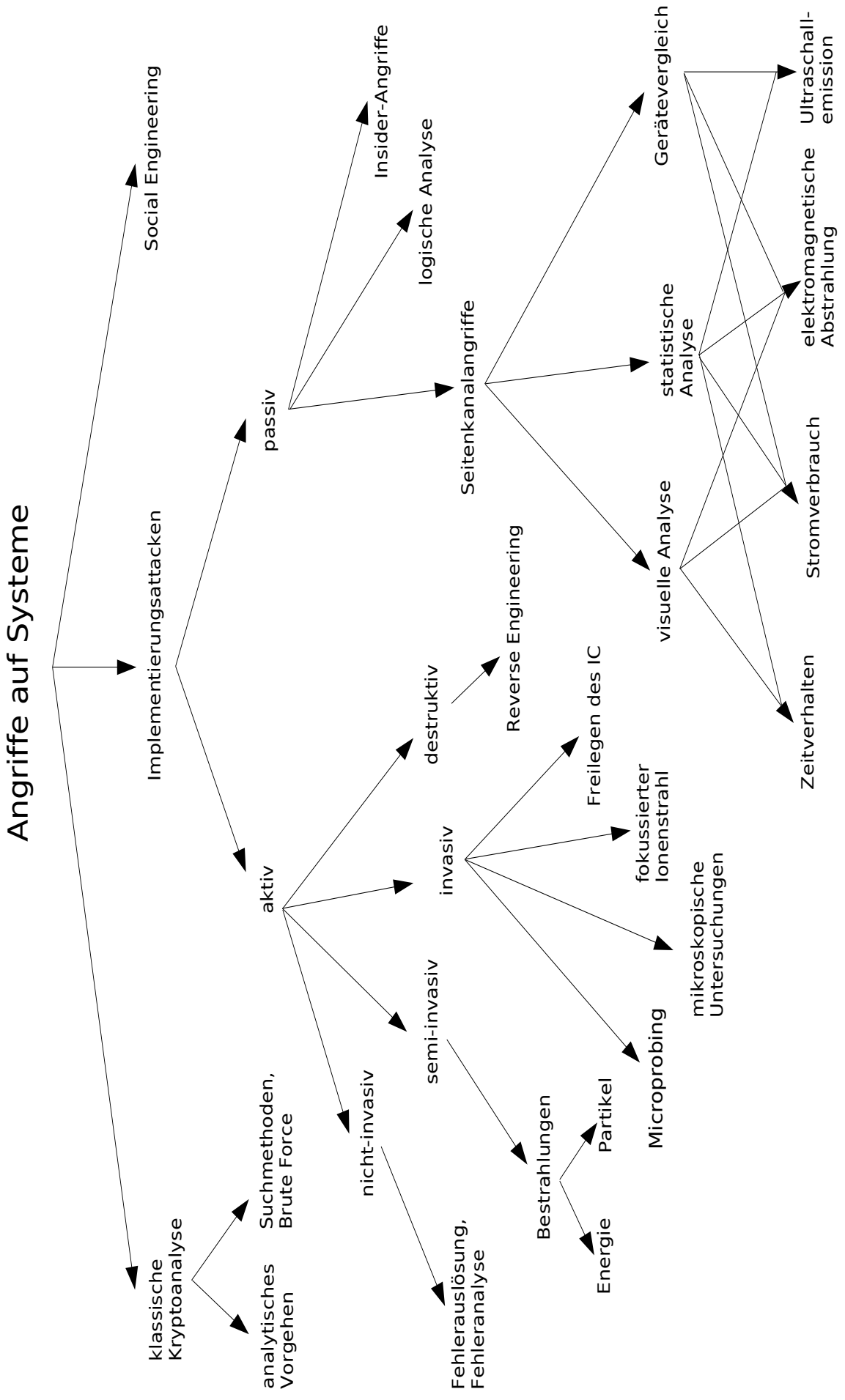

Abbildung 6.2: Angriffe auf Systeme

Der Begriff *Social Engineerig* bezeichnet das „Erlangen vertraulicher Informationen durch Annäherung an Geheimnisträger mittels gesellschaftlicher Kontakte“[4]. Es dient dazu, eine Zielperson dazu zu verleiten, vertrauliche Informationen preiszugeben, auf die der Täuschende auf offiziellem Wege keinen Zugriff erhalten würde. Sie beruht häufig auf der Ausnutzung informeller sozialer Strukturen und immanenter psychologischer Verhaltensmuster. So werden beispielsweise Mitarbeiter einer Firma oder Privatpersonen auf Internetseiten oder durch E-Mails unter einem Vorwand aufgefordert, Passwörter oder persönliche Identifikationsnummern einzugeben (sog. *Phishing*), die ein Angreifer dann abfängt und anschließend für seine Zwecke mißbraucht.

Implementierungsattacken sind eine Gruppe von Angriffen auf kryptographische Verfahren, die nicht die theoretischen Schwächen oder das menschliche Fehlverhalten, sondern die physikalischen Eigenschaften der Implementierung dieser Verfahren bei der Ausführung auf einem Mikroprozessor oder in einem Chip ausnutzen. Ein solcher Angriff kann entweder aktiv oder passiv erfolgen. Ersteres ist der Fall, sobald an dem Gerät, das die kryptographische Anwendung ausführt, Manipulationen vorgenommen werden. Die Manipulation kann dabei *invasiv* oder *nicht-invasiv*, also „eindringend“ oder „nicht-eindringend“ sein.

Ein aktiver und *invasiver* Angriff ist beispielsweise das Freilegen des Chips um anschließend mit Hilfe von Meßspitzen einzelne Leitungen anzuzapfen (engl. microprobing) und die darüber transportierten geheimen Daten zu enthüllen. Auch das Auslesen einzelner Speicherzellen, um den intern gespeicherten geheimen Schlüssel auszulesen fällt in diese Kategorie. Diese Angriffe werden auch als *Penetration* (dt. Eindringen) bezeichnet. Im Gegensatz zu einem *destruktiven* Angriff bleibt der Chip bei dieser Angriffsform jedoch weiterhin funktionstüchtig.

Dies ist bei der Methode des *Reverse-Engineerings*[5] nicht der Fall. Sie ist eine Analysemethode um aus einem bestehenden, fertigen System oder einem meistens industriell gefertigten Produkt durch Untersuchung der Strukturen, Zustände und Verhaltensweisen, die Konstruktionselemente zu extrahieren. Aus dem fertigen Objekt wird somit wieder ein Plan gemacht, anhand dessen eine Kopie des Objekts angefertigt werden kann.

Im Gegensatz zu invasiven Angriffen sind *nicht-invasive* Angriffe als solche nicht zu erkennen, da hierbei keine äußerlich sichtbaren Manipulationen an dem Gerät oder Chip vorgenommen werden. Dies sind beispielsweise *Fehleranalysen* (engl. fault analysis), bei denen während der Berechnung durch Einflüsse von außen Fehler induziert werden, sodaß das Endergebnis unerwünschterweise Hinweise auf geheimzuhaltende Daten aufweist. Entsprechende Störungen können durch Schwankungen der Strom- oder Spannungsversorgung (engl. glitches) oder der Taktrate, aber auch z.B. durch Bestrahlung des Chips oder Temperaturschwankungen erreicht werden [Cryp06].

Bei den *logischen* Angriffen, die ebenfalls zu den passiven Implementierungsattacken gehören, werden Fehler auf Software- oder Protokollebene, sogenannte *Bugs* ausgenutzt. Dabei ist kein Zugriff auf das Gerät notwendig, der Angriff findet ausschließlich auf Softwareebene statt.

[4] http://de.wikipedia.org/wiki/Social_Engineering_%28Sicherheit%29
[5] http://de.wikipedia.org/wiki/Reverse_Engineering

6.3.1 Seitenkanalangriffe

Die bedeutendste Form von Implementierungsattacken sind sogenannte *Seitenkanalangriffe*. Sie sind ebenfalls nicht-invasiv, da hierbei keine Manipulationen an den Geräten vorgenommen werden und außerdem passiv, da sie lediglich auf Beobachtungen und Messungen bestimmter physikalischer Eigenschaften beruhen. Diese Angriffe werden daher auch als *Monitoring* (dt. Beobachtung) bezeichnet.

Sie sind eine mächtige Angriffsform und stellen eine ernsthafte Bedrohung der Sicherheit kryptographischer Module dar, da zu ihrer Durchführung keine teure Ausrüstung oder spezielle Kenntnisse notwendig sind und sie mit verhältnismäßig geringem Aufwand durchführbar sind. Ein Angreifer muß dabei lediglich Zugriff auf das Gerät haben, in dem die Verschlüsselung stattfindet. Ziel solcher Angriffe sind beispielsweise *Smartcards*, auf denen kryptographische Anwendungen laufen. Sie sind heutzutage weit verbreitet und kommen unter anderem als Geldkarten, Krankenversicherungskarten und Gebäudezugangskarten zum Einsatz.

Seitenkanalangriffe nutzen physikalische Eigenschaften aus, die während der Durchführung kryptographischer Operationen auf einem Prozessor oder in einer Smartcard auftreten. Während ein kryptographischer Algorithmus als Programm auf einem Prozessor ausgeführt wird, ist der geheime Schlüssel als physikalisches Objekt im Prozessor gespeichert. In einigen Programmteilen wird dieser geheime Schlüssel mit der Eingabe verknüpft um die Ausgabe zu produzieren. In diesen schlüsselabhängigen Programmteilen treten physikalische Eigenschaften des Prozessors auf, die deterministisch vom Schlüssel abhängen [LWBG01]. Diese Eigenschaften sind z.B. die Ausführungszeit von Berechnungen, der Stromverbrauch oder die elektromagnetische Abstrahlung. Sie werden auch als *Informationslecks* der Geräte bezeichnet.

Durch Messung dieser physikalischen Eigenschaften kann dann ein physikalischer Kommunikationskanal aufgebaut werden. Falls ein solcher *Seitenkanal* eines Kryptosystems Informationen über den geheimen Schlüssel überträgt, kann er zur Kryptoanalyse benutzt werden. Dieses Prinzip ist schematisch in Abb. 6.3 dargestellt.

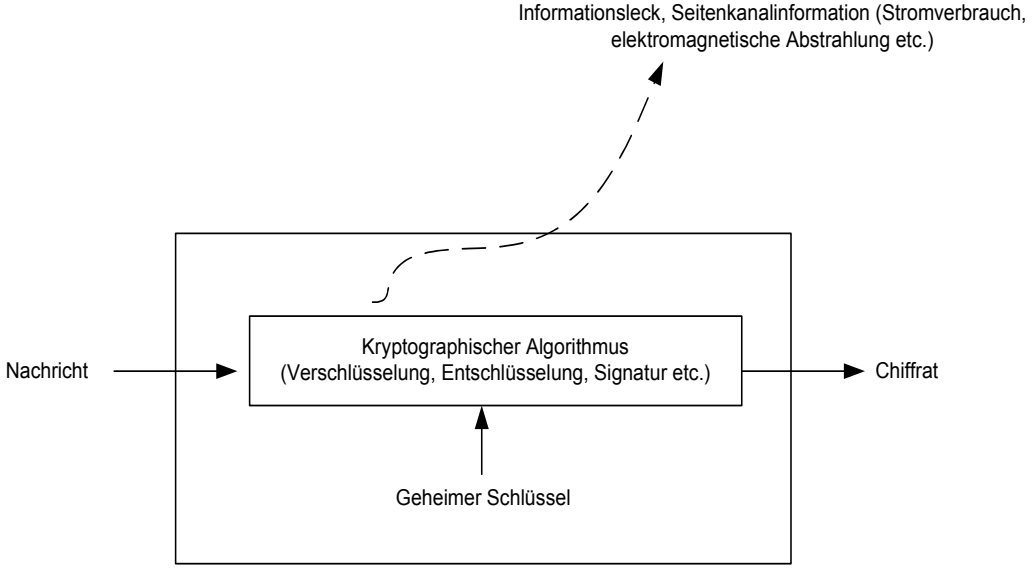

Abbildung 6.3: Prinzip der aus einem Seitenkanal entweichenden Information

Treten während der durchgeführten Berechnungen Zwischenergebnisse auf, die deterministisch vom Schlüssel abhängen und ist dadurch auch die Seitenkanalinformation abhängig vom geheimen Schlüssel, so kann diese Information zur Rekonstruktion des geheimen Schlüssels genutzt und dadurch das Verfahren gebrochen werden.

Bei einem Seitenkanalangriff werden somit Informationen gesammelt, die während der Programmausführung entstehen. Man erhält nicht nur Informationen über das Ergebnis der Berechnung, sondern auch über die eigentliche Ausführung, d.h. die internen Berechnungen und die während dieser Berechnungen auftretenden Zwischenergebnisse.

Die meisten Implementierungen kryptographischer Verfahren, die keine Gegenmaßnahmen besitzen, sind anfällig für Seitenkanalangriffe. Da hierbei nicht die mathematischen Schwächen eines Kryptoalgorithmus ausgenutzt werden, sondern die physikalischen Eigenschaften der praktischen Umsetzung des Algorithmus, sind auch viele theoretisch (d.h. mathematisch) als sicher angesehene Verfahren auf diese Weise angreifbar.

6.3.1.1 CMOS-Technologie

Die am häufigsten durchgeführten Seitenkanalangriffe sind die Analyse des Stromverbrauchs sowie der elektromagnetischen Abstrahlung eines Gerätes. Ein Grund hierfür ist, daß die praktische Umsetzung kryptographischer Algorithmen heute meist durch digitale Schaltungen in *CMOS*[6]-Technologie erfolgt. Der Stromverbrauch sowie die elektromagnetische Abstrahlung dieser integrierten Schaltkreise hängen sehr stark von den verarbeiteten Daten ab und enthüllen auf diese Weise auch Informationen über den geheimen Schlüssel [Mang04].

CMOS-Schaltkreise bestehen aus *Logikgattern*, die durch Leitungen verbunden sind. Je nach Ladungszustand repräsentieren diese Leitungen eine logische 1 oder eine logische 0: eine mit der Masse verbundene Leitung repräsentiert eine 0 (LOW), eine mit der Stromversorgung verbundene Leitung eine 1 ($HIGH$). Ein solches Logikgatter hat ein oder mehrere Eingangssignale und erzeugt daraus durch logische Operationen ein Ausgangssignal. Es besteht aus *p-MOS-* und *n-MOS-Transistoren*. Diese sind schematisch in Abb. 6.4 dargestellt.

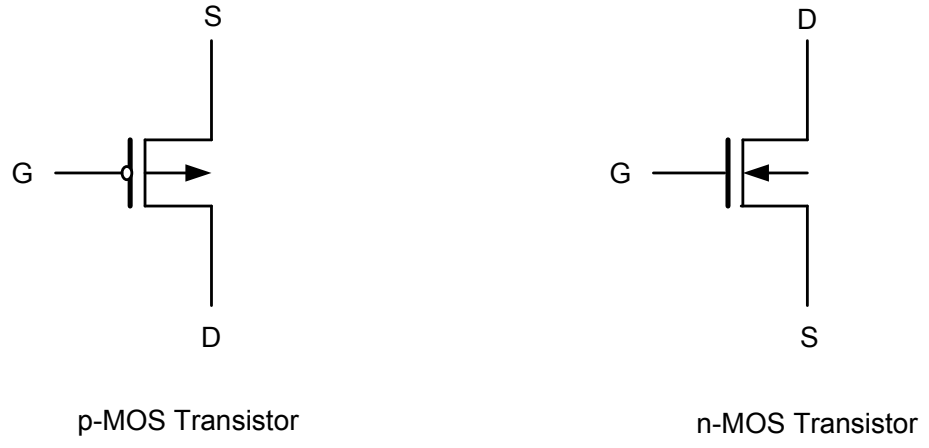

Abbildung 6.4: Schematische Darstellung eines p-MOS- und eines n-MOS-Transistors

[6]Complementary Metal Oxide Semiconductor

Ein p-MOS-Transistor wird auch als selbstsperrend bezeichnet. Er leitet genau dann eine Verbindung zwischen der *Quelle* (engl. source) S und der *Senke* (engl. drain) D durch, wenn an seinem *Gate* G eine logische 0 anliegt. Andernfalls ist der Transistor gesperrt. Die Quelle ist dabei bei einem p-MOS Transistor meist mit der Spannungsversorgung verbunden. Ein n-MOS-Transistor verhält sich komplementär zu einem p-MOS-Transistor: er schaltet bei einer 1 am Gate eine Verbindung zwischen Quelle und Senke. Die Quelle ist dabei mit der Masse verbunden. Er wird auch als selbstleitend bezeichnet.

Das einfachste Logikgatter in CMOS-Technologie ist ein Inverter. Er besteht aus genau einem p-MOS und einem n-MOS-Transistor wie in Abb. 6.5 dargestellt. Da sein charakteristischer Stromverbrauch repräsentativ für alle anderen CMOS-Gatter ist, genügt es, seine Seitenkanalinformation zu untersuchen, um Informationslecks von CMOS-Schaltkreisen im Allgemeinen zu analysieren.

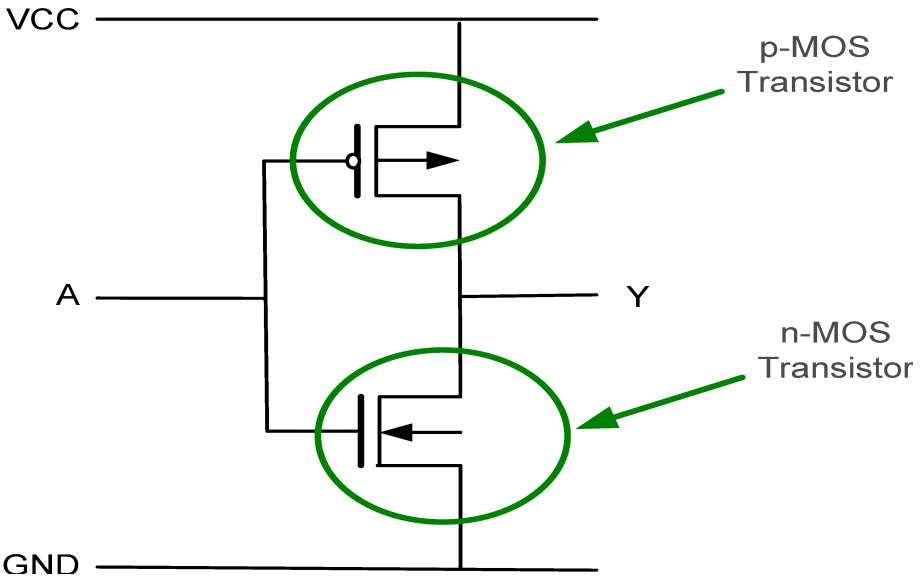

Abbildung 6.5: Schematische Darstellung eines Inverter in CMOS-Technologie

Liegt am Eingang A des Inverters eine logische 0 an, so leitet der p-MOS-Transistor und der n-MOS-Transistor sperrt. Der Ausgang Y wird also über den p-MOS-Transistor mit der Spannung VCC verbunden. Dadurch ist der Ausgangspegel Y des Inverters *HIGH*, d.h. er repräsentiert eine logische 1. Im umgekehrten Fall, d.h. für eine Eingangsspannung von 1 am Eingang A, sperrt der p-MOS-Transistor und der n-MOS-Transistor leitet. Dadurch gibt es eine Verbindung von der Masse GND zum Ausgang Y. Die Spannung am Ausgang ist also auch 0V (LOW), entsprechend dem logischen Wert 0. Analog dazu wird auch bei komplexeren Logikgattern aus den Eingangssignalen ein Ausgangssignal erzeugt.

Sind die Eingangssignale der Transistoren und damit auch die des entsprechenden Gatters stabil, d.h. es findet kein Signalwechsel statt, so ist der Spannungspegel in diesem Fall entweder *HIGH* oder LOW und die in diesem Ruhezustand auftretenden *Leckströme* sind vernachlässigbar gering. Tritt jedoch bei einem Gatter ein Umschaltvorgang, d.h. ein Zustandswechsel von 0 nach 1 oder von 1 nach 0 auf, so sind kurzzeitig beide Transistoren (p-MOS und n-MOS) leitend, und es fließt für eine kurze Zeit ein Strom zwischen der Masse (GND) und der Spannungsversorgung

(VCC) der Transistoren. Dieser Stromfluß ist signifikant größer als die Leckströme und kann gemessen und zur Kryptoanalyse genutzt werden.

Wesentlich ist dabei, daß alle Umschaltvorgänge der Gatter zur selben Zeit, bei einer Taktflanke, passieren. Je mehr Schaltungselemente gleichzeitig ihren Zustand ändern, desto mehr Strom wird aufgenommen. Der Gesamtstromverbrauch einer Schaltung oder eines Hardwaremoduls in CMOS-Technologie ist daher abhängig vom Stromverbrauch der einzelnen Gatter. Bei einem idealisierten Modul beträgt der Gesamtstromverbrauch $P_i(t)$ zum Zeitpunkt t: $P_i(t) = \sum_i f_i(t)$, wobei $f_i(t)$ der Stromverbrauch eines Gatters bei einem Umschaltvorgang ist [AiOs00]. Da die Zustandswechsel der Gatter wiederum von den verarbeiteten Daten und dem verwendeten geheimen Schlüssel abhängen, gibt es Korrelationen zwischen diesen Daten und dem Stromverbrauch. Diese lassen sich für einen Angriff durch einfache oder differentielle Analyse des Stromverbrauchs einer Schaltung ausnutzen.

6.3.1.2 Einfache Stromverbrauchsanalyse

Bei einer *einfachen Stromverbrauchsanalyse* (engl. simple power analysis, SPA) wird eine einzelne Stromverbrauchskurve eines Gerätes oder einer Schaltung in CMOS-Technologie betrachtet, die eine Ver- oder Entschlüsselung mit einem geheimen Schlüssel durchführt. Anhand des Verlaufs dieser Kurve versucht man dann, Informationen über die verarbeiteten Daten und den verwendeten geheimen Schlüssel zu gewinnen.

Man geht dabei davon aus, daß die Stromaufnahme zum Zeitpunkt t sehr stark von den gerade ausgeführten Operationen und den Eingangsdaten abhängt. Wenn solche Operationen, wie zum Beispiel Additionen oder bitweise logische Operationen, eine sehr charakteristische Stromaufnahme haben, dann ist es einem Angreifer möglich, aufgrund solcher Charakteristiken zwischen den einzelnen Operationen zu unterscheiden. Hängen diese (bzw. deren Abfolge) wiederum vom verwendeten (geheimen) Schlüssel ab, so kann man Teile dieses Schlüssels oder den gesamten Schlüssel durch eine visuelle Inspektion der Messung identifizieren [AiOs00].

Ein Beispiel für einen solchen erfolgreichen Angriff ist die *modulare Exponentiation* $x^y \pmod p$ wie sie im RSA-Algorithmus zum Einsatz kommt. Diese wird häufig durch ein *Square-and-Multiply*-Verfahren zum schnellen Potenzieren realisiert. Dabei wird zunächst der Exponent y, der in diesem Fall der geheime Schlüssel ist, in Binärdarstellung gebracht. Beginnend mit dem höchstwertigen Bit erfolgt nun die schrittweise Abarbeitung des Exponenten. Je nachdem ob das gerade verarbeitete Bit eine 0 oder eine 1 ist, wird entweder nur quadriert oder quadriert und anschließend multipliziert. Wenn es möglich ist Multiplikation und Quadrierung anhand ihres Stromverbrauchs zu unterscheiden, so kann die Binärdarstellung des Exponenten und damit der geheime Schlüssel anhand der Stromverbrauchskurve rekonstruiert werden.

Paul Kocher et al. führten in [KoJJ99] eine einfache Stromverbrauchsanalyse einer Smartcard durch, die einen DES ausführte. Dies ergab die in Abb. 6.6 dargestellte Stromverbrauchskurve. Dort sind deutlich die 16 einzelnen Runden einer Ausführung von DES zu erkennen.

Erhöht man die Abtastrate bei der Messung und betrachtet nur einen kleinen Ausschnitt der gesamten Verschlüsselungsoperation, so lassen sich anhand der Kurve

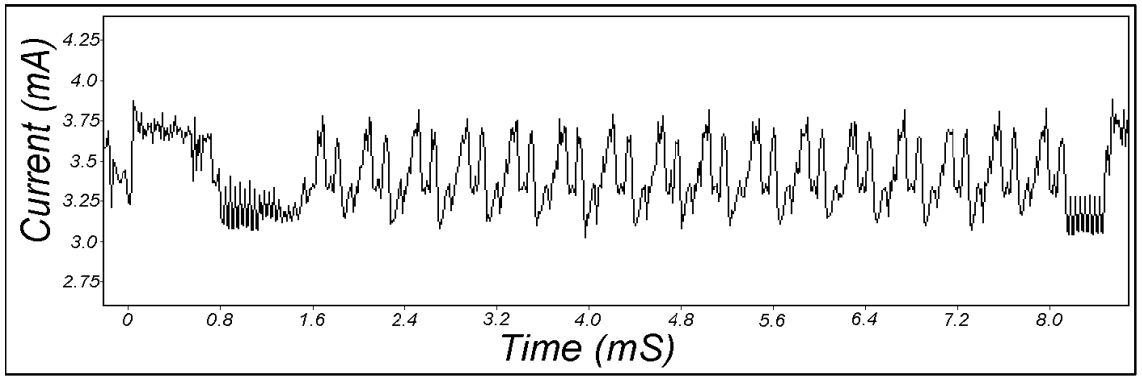

Abbildung 6.6: Stromverbrauchskurve einer Ausführung von DES

noch wesentlich mehr Details der Berechnung erkennen. Abb. 6.7 zeigt eine detailliertere Ansicht derselben Stromverbrauchskurve mit einer höheren Auflösung und der Betrachtung von lediglich zwei der 16 Runden. Hier dargestellt sind die zweite und dritte Runde der Verschlüsselungsoperation. Man kann hier die unterschiedliche Anzahl der Verschiebungen der Schlüsselbits in den verschiedenen Runden erkennen. In der zweiten Runde ist dies ein Bit (siehe Pfeil links), in der dritten Runde zwei (siehe Pfeile rechts).

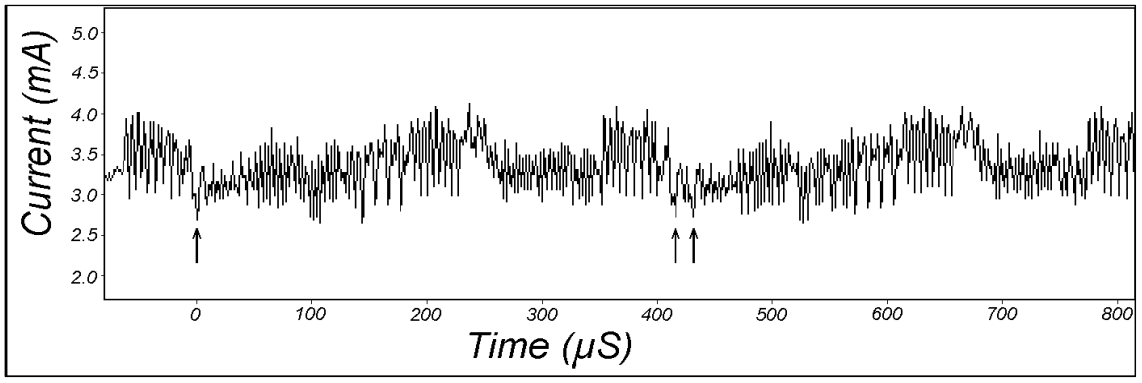

Abbildung 6.7: Stromverbrauchskurve mit höherer Auflösung

Die Voraussetzung für einen solchen Angriff ist, daß dem Angreifer der genaue Ablauf des Programms und Details der Implementierung sowie der genaue Ausführungszeitpunkt der betrachteten schlüsselabhängigen Operationen bekannt sind.

Angriffe mit einfacher Stromverbrauchsanalyse können jedoch relativ leicht verhindert werden, beispielsweise durch Einfügen von Dummy-Operationen, durch Randomisierung der Ausführungsreihenfolge der Operationen oder durch die Vermeidung datenabhängiger Sprünge und Verzweigungen im Programm. Ziel solcher Gegenmaßnahmen ist es dabei stets, einen datenunabhängigen Stromverbrauch zu erreichen, sodaß der Verlauf der Stromverbrauchskurve nicht mehr von den verarbeiteten Daten und dem verwendeten geheimen Schlüssel abhängig ist.

6.3.1.3 Differentielle Analyse mit Korrelationen

Wesentlich schwieriger zu verhindern als eine einfache Analyse sind dagegen Angriffe mit *differentieller Analyse* des Stromverbrauchs (engl. differential power analysis, DPA) oder der elektromagnetischen Abstrahlung. Dabei werden Korrelationen

zwischen den verarbeiteten Daten und dem Stromverbrauch bzw. der Abstrahlung während dieser Verarbeitung ausgenutzt.

Im Gegensatz zur einfachen Analyse wird hier nicht nur eine einzelne, sondern sehr viele (in der Praxis mehrere tausend) Messungen des Stromverbrauchs oder der Abstrahlung eines Gerätes bei der Durchführung vieler Ent- oder Verschlüsselungen mit demselben Schlüssel und verschiedenen Eingangsdaten betrachtet. Der Angriff konzentriert sich dabei jeweils nur auf ein Teilergebnis in einer der Runden sowie einen Teil des geheimen Schlüssels. Im Gegensatz zur einfachen Analyse muß ein Angreifer hier nicht so viele Implementierungsdetails des Algorithmus kennen.

Der Ablauf eines solchen *Korrelationsangriffs* ist dann wie folgt:
Es werden n Messungen des Stromverbrauchs bzw. der elektromagnetischen Abstrahlung eines Gerätes gemacht, das n verschiedene Eingangsdaten mit demselben Schlüssel ver- oder entschlüsselt. Dabei ist n beliebig groß, in der Praxis sind dies meist mehrere tausend Durchgänge. Voraussetzung dabei ist, daß die Eingangsdaten (Klartexte oder Chiffrate) bekannt sind, was jedoch im Allgemeinen der Fall ist, da ein Angreifer Zugriff auf das Gerät und eine ausreichend große Anzahl Klartexte oder Chiffrate hat.

Der Angriff konzentriert sich nun auf einen Teil eines Zwischenergebnisses, das in einer der Runden mit einem Teil des geheimen Schlüssels berechnet wird. Wichtig ist dabei, daß der betrachtete Rundenschlüssel kurz ist oder daß er in kurzen Teilblöcken verarbeitet wird. Im Fall eines Angriffs auf DES könnte das betrachtete Teilergebnis beispielsweise die Ausgabe der ersten S-Box in der ersten Runde bei der Verschlüsselung sein. Da die Verschlüsselungsfunktion bekannt ist, kann genau rekonstruiert werden, welche Bits der Eingangsdaten und welche Schlüsselbits an der Berechnung dieser Ausgabe beteiligt sind. In diesem Fall sind dies 6 Schlüsselbits, sodaß es also $2^6 = 64$ Möglichkeiten für den verwendeten Teilschlüssel gibt.

Die Funktion, mit der die Ausgabe der betrachteten S-Box berechnet wird, wird als *Selektionsfunktion* bezeichnet. Mit Hilfe dieser Selektionsfunktion kann für alle n verschiedenen Eingangsdaten und jeden der 64 möglichen Schlüssel der Wert der betrachteten S-Box berechnet werden.

Wie in Abschnitt 6.3.1.1 beschrieben, ist der Stromverbrauch einer Schaltung in CMOS-Technologie abhängig von den verarbeiteten Daten und somit auch vom verwendeten geheimen Schlüssel. Der Gesamtstromverbrauch der Schaltung zu einem bestimmten Zeitpunkt während der Ausführung ist dabei proportional zur Anzahl der Umschaltvorgänge der Gatter zu diesem Zeitpunkt. Ebenso ist auch die elektromagnetische Abstrahlung, die beim Umschalten produziert wird, proportional zu den verarbeiteten Daten.

Man untersucht nun für jeden der 64 möglichen Schlüssel Korrelationen zwischen den mit diesem Schlüssel berechneten Werten der Selektionsfunktion für die verschiedenen Eingangsdaten und den Stromverbrauchs- oder Abstrahlungsmessungen. Die Messungen werden dabei abhängig vom Wert der Selektionsfunktion bzw. dessen Hamminggewicht unterschiedlich gewichtet und anschließend addiert.

Nur bei der richtigen Schlüsselhypothese, die dem tatsächlich zur Berechnung verwendeten geheimen Schlüssel entspricht, ergibt sich durch die Addition aller Kurven an der Stelle, an der die Selektionsfunktion berechnet wird, ein signifikanter Ausschlag. In allen anderen Fällen, bei denen zur Berechnung der Selektionsfunktion ein

falscher Schlüsselwert verwendet wurde, korreliert der Stromverbrauch bzw. die Abstrahlung während der Berechnungen nicht mit den Werten der Selektionsfunktion und es ergibt sich kein solcher Ausschlag.

Andere Beschreibungen differentieller Stromverbrauchsanalysen mit Korrelationen finden sich unter anderem in [KoJJ99] und [AiOs00].

Nach dem Bekanntwerden dieser Angriffsmöglichkeit auf kryptographische Systeme wurden verschiedene Gegenmaßnahmen auf Hardware- und Softwareebene entwickelt, um Implementierungen kryptographischer Algorithmen resistent gegen diese Form von Angriffen zu machen.

6.4 Maßnahmen gegen Seitenkanalangriffe

Die Angriffsmöglichkeiten auf kryptographische Verfahren hängen stark von ihrer jeweiligen Implementierung ab. Erfolgt diese beispielsweise als Programm in Software, das von einem Universalprozessor ausgeführt wird, so ergeben sich andere Angriffsmöglichkeiten als bei einer Implementierung in Hardware, d.h. in einem Chip oder auf einem Mikroprozessor. Dementsprechend gibt es auch unterschiedliche Methoden um ein solches System vor den entsprechenden Angriffen zu schützen. Sie werden als *Gegenmaßnahmen* (engl. countermeasures) bezeichnet.

Die Implementierung von Gegenmaßnahmen gegen Angriffe ist dabei fast immer mit zusätzlichen Kosten in Bezug auf die Geschwindigkeit der Ausführung (Performanz), den Rechenaufwand, die Chipfläche oder den Energieverbrauch verbunden. Außerdem erhöht sich der Aufwand für das Design und die Entwicklung.

Um einen Kompromiß zwischen der Resistenz gegen Angriffe und den damit verbundenen höheren Kosten zu finden, kombinieren die Entwickler kryptographischer Module daher meist verschiedene Gegenmaßnahmen. Außerdem werden je nachdem, ob es sich um eine Implementierung in Hardware oder Software handelt und in welcher Technologie diese erfolgt, unterschiedliche Schutzmaßnahmen gewählt und in die Programme und Geräte eingebaut. Abb. 6.8 gibt einen Überblick über die verschiedenen Gegenmaßnahmen.

6.4.1 Protokollebene

Eine Maßnahme gegen Angriffe auf Protokollebene ist die Verwendung kurzlebiger (engl. ephemeral) Schlüssel, die nur für eine geringe Anzahl kryptographischer Operationen verwendet und danach ausgetauscht werden [Mang04]. Dadurch ist ein Angreifer nicht mehr in der Lage, genügend Messungen zu sammeln um Korrelationen zwischen diesen Messungen für eine differentielle Analyse auszunutzen.

Bei dieser Methode wird allerdings nicht die Korrelation zwischen dem Stromverbrauch und den während der Berechnung auftretenden und vom geheimen Schlüssel abhängigen Zwischenergebnissen reduziert. So wäre beispielsweise eine einfache Stromverbrauchsanalyse (SPA) in diesem Fall weiterhin möglich. Außerdem müssen dafür in dem zu schützenden kryptographischen Modul ständig neue Schlüssel generiert werden. In der Praxis erfolgt dies meist mit Hilfe eines *Masterkeys*. Dieser kann von einem Angreifer möglicherweise aus der entweichenden Seitenkanalinformation gewonnen werden und muß daher zusätzlich gesondert geschützt werden.

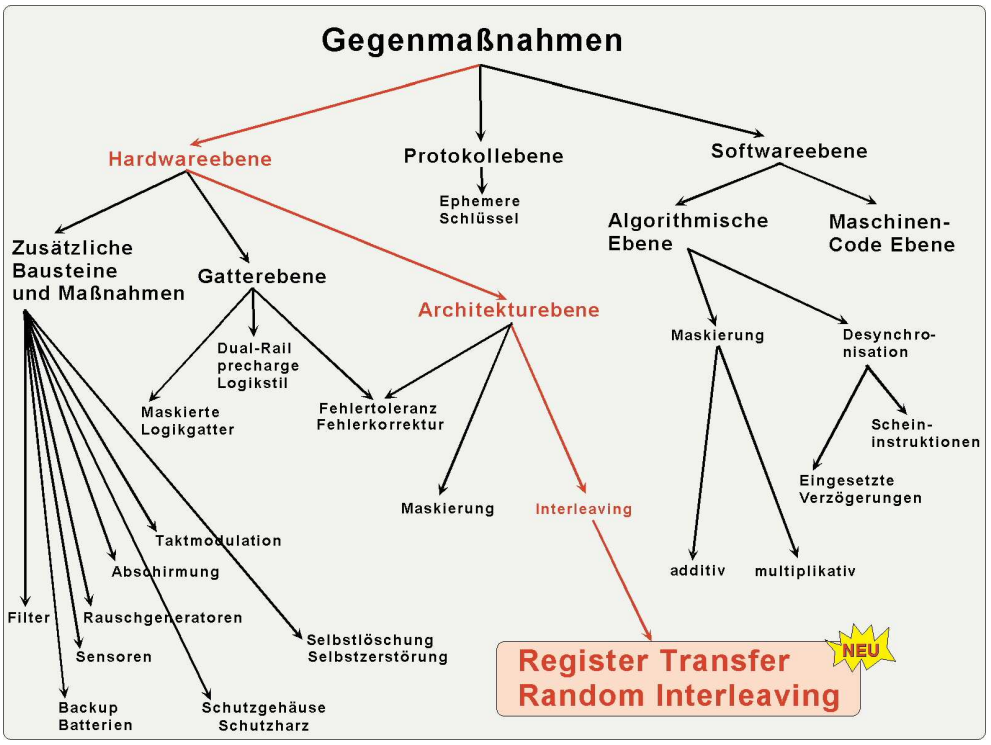

Abbildung 6.8: Maßnahmen gegen Angriffe

6.4.2 Hardwareebene

Auf Hardwareebene kann man durch entsprechende Maßnahmen erreichen, daß entweder gar keine Seitenkanalinformation mehr entweicht, oder daß die entweichende Information (wie z.B. der Stromverbrauch) unabhängig von den gerade verarbeiteten Daten und dem verwendeten geheimen Schlüssel ist. Ersteres erreicht man beispielsweise durch Abschirmung des Gerätes oder durch die Verwendung von Schutzgehäusen, durch die keine Information über die intern verarbeiteten Daten mehr entweichen kann.

Um Angriffe durch Induzierung von Fehlern zu detektieren und zu verhindern, werden häufig Sensoren in die Geräte eingebaut, die eine Veränderung der Stromversorgung, der Temperatur oder des Taktes erkennen. In einem solchen Fall wird das Gerät dann zurückgesetzt, gelöscht oder zerstört.

Eine weitere Möglichkeit ist das Verfälschen der entweichenden (Seitenkanal-)Information durch Einfügen von künstlichem Rauschen oder Filtern, die das Signal-Rausch-Verhältnis (engl. signal-to-noise ratio, SNR) verschlechtern, sodaß ein Angreifer das nützliche Signal, das unter Umständen Informationen über den geheimen Schlüssel preisgibt, nicht mehr von dem zusätzlich eingefügten Rauschen trennen kann.

6.4.3 Gatterebene

Um zu verhindern, daß die physikalischen Eigenschaften der Implementierung eines kryptographischen Verfahrens für einen Angriff genutzt werden können, versucht man, eine *Dekorrelation* zwischen den verarbeiteten Daten und der dabei aus einem Seitenkanal entweichenden Information zu erreichen. Das Ziel ist dabei, diese Information, wie z.B. den Stromverbrauch oder die elektromagnetische Abstrahlung,

unabhängig von den Daten und dem verwendeten geheimen Schlüssel zu machen, sodaß diese nicht mehr zu Kryptoanalyse genutzt werden können.

Der Stromverbrauch einer Schaltung in CMOS-Technologie ist abhängig von dem ihrer elementarsten Teile, der Gatter (vgl. Kapitel 6.3.1.1). Findet am Ausgang eines solchen Gatters eine Transition, d.h. ein Übergang von 0 nach 1 oder von 1 nach 0 statt, so wird hierfür Energie verbraucht oder freigegeben. Für die Übergänge $0 \rightarrow 0$ und $1 \rightarrow 1$, d.h. den Ruhezustand, wird keine Energie verbraucht.

Der Stromverbrauch der Gatter und damit auch der gesamten Schaltung ist also nicht nur vom Wert des Ausgangssignals, sondern auch von seinem Wechsel, d.h. von der Reihenfolge seiner Werte, dem *Hamming-Abstand*, abhängig. Hängt diese Reihenfolge wiederum vom geheimen Schlüssel ab, so kann diese Korrelation zwischen geheimem Schlüssel und Energieverbrauch der Schaltung für einen Angriff genutzt werden.

Um auf Gatterebene eine Dekorrelation zu erreichen, muß daher ein Logikstil verwendet werden, bei dem der Energieverbrauch der Gatter und damit auch der der gesamten Schaltung in jedem Takt gleich und unabhängig von den Eingangsdaten oder deren Hamming-Distanz ist [Mang04].

6.4.3.1 Differentielle Logik

Eine Möglichkeit den Stromverbrauch eines Gatters und damit den der gesamten Schaltung unabhängig von den Eingangssignalen und den auftretenden Signalübergängen zu machen, ist die Verwendung *differentieller* oder *Dual-Rail*-Logik. Dabei werden jeweils zwei Leitungen verwendet um die logischen Werte 0 und 1 zu repräsentieren. Die Zustände 01 bzw. 10 eines solchen Leitungspaars entsprechen der logischen 0 bzw. 1. Die Bedeutung der anderen beiden Zustände 00 und 11 ist bei den unterschiedlichen Dual-Rail Logikstilen verschieden.

Bei der differenziellen Logik wird für jedes Eingangssignal in_i eines Gatters, auch dessen Komplement $\overline{in_i}$ verwendet. Mit Hilfe der *De Morganschen Gesetze*[7] kann damit zusätzlich zum Ausgang q des Gatters auch dessen Komplement $\overline{q}$ erzeugt werden.

Wenn das Ausgangssignal q eines Gatters aufgrund einer Änderung der Eingangssignale einen Signalwechsel von 0 nach 1 oder von 1 nach 0 macht, so wird dabei die Energie $E_{0\rightarrow1}$ oder $E_{1\rightarrow0}$ verbraucht. Der Ausgang $\overline{q}$ macht dabei stets den komplementären Wechsel und verbraucht dabei die Energiemenge des komplementären Übergangs $E_{1\rightarrow0}$ oder $E_{0\rightarrow1}$. Der gesamte Energieverbrauch einer solchen Schaltung ist daher immer $E_{0\rightarrow1} + E_{1\rightarrow0}$ unabhängig davon, welcher der Ausgänge q und $\overline{q}$ welchen der beiden Signalübergänge ($0 \rightarrow 1$ oder $1 \rightarrow 0$) gemacht hat.

Unter der Voraussetzung, daß ein Umschaltvorgang von q dieselbe Menge an Strom verbraucht wie ein Umschalten von $\overline{q}$, ist der Gesamtstromverbrauch der Schaltung also unabhängig von den verarbeiteten Daten. Um dies zu gewährleisten ist es notwendig, daß die kapazitäre Last (engl. capacitive load) der beiden Leitungen gleich ist. Dadurch ist bei der Verwendung von differentieller Logik sichergestellt, daß bei einem Signalwechsel immer dieselbe Energiemenge verbraucht wird.

[7] $\overline{a \wedge b} = \overline{a} \vee \overline{b}$,
$\overline{a \vee b} = \overline{a} \wedge \overline{b}$

Nachteil dieses Logikstils ist jedoch, daß doppelt so viele Leitungen und Gatter und damit doppelt so viel Fläche für die Schaltung benötigt werden wie bei der Single-Rail Technik. Außerdem wird lediglich der Unterschied im Stromverbrauch zwischen den Signalübergängen $0 \rightarrow 1$ und $1 \rightarrow 0$ beseitigt. Der Unterschied zwischen einem Signalwechsel ($0 \rightarrow 1$ oder $1 \rightarrow 0$) und keinem Wechsel, d.h. dem Ruhezustand, bleibt bestehen.

6.4.3.2 Precharge-Logik

Um zu erreichen, daß der Stromverbrauch eines Gatters bei einem Wechsel des Ausgangssignals, d.h. einem Übergang von 0 nach 1 oder von 1 nach 0 gleich dem Ruhezustand des Gatters ist, kann ein sogenannter *Precharge*-Logikstil verwendet werden. Dabei wird jeder Takt in zwei Phasen eingeteilt: eine *Precharge-* und eine *Evaluationsphase*. In der Prechargephase werden alle dualen Leitungspaare auf den Wert 00 oder 11 und damit auf einen definierten Spannungswert gesetzt. Erst in der zweiten Hälfte des Taktes, der Evaluationsphase, wird dann der logische Wert 0 oder 1 gespeichert, indem die dualen Leitungen die Werte 01 oder 10 annehmen. Zu Beginn des nächsten Taktes werden sie dann wieder beide auf 0 oder 1 gesetzt, unabhängig davon, ob im nächsten Takt ein Wechsel des Ausgangssignals stattfindet oder nicht. Dadurch finden in jedem Takt exakt zwei Signalübergänge statt.

In der Precharge-Phase werden beide Leitungen q und $\overline{q}$ auf denselben Wert, also z.B. beide auf 0 gesetzt. Anschließend führt entweder q oder $\overline{q}$ einen Signalwechsel von 0 nach 1 aus und wird danach wieder auf 0 gesetzt. Das Umschalten zwischen den beiden Phasen wird dabei durch den Takt kontrolliert. Im Allgemeinen wird die Precharge-Phase mit der fallenden und die Evaluationsphase mit der steigenden Flanke getaktet. Unter der Voraussetzung, daß das Umschalten beider Leitungen, d.h. von q und $\overline{q}$ dieselbe Menge an Energie verbraucht, ist der Gesamtstromverbrauch des Gatters und damit der gesamten Schaltung in jedem Takt gleich. Es wird jeweils die Energie $E_{0\rightarrow1} + E_{1\rightarrow0}$ verbraucht.

Auch wenn sich die Eingangssignale und damit der Wert des Ausgangssignals eines Gatters zwei oder mehr Takte lang nicht ändern, finden dennoch zwei Signalwechsel pro Takt statt. Somit läßt sich anhand der Stromverbrauchskurve kein Unterschied mehr zwischen einem Wechsel des Ausgangssignals, d.h. einem Umschaltvorgang des Gatters, und seinem Ruhezustand erkennen. Es wird also eine Dekorrelation zwischen dem Stromverbrauch und den verarbeiteten Daten erreicht.

6.4.4 Architekturebene

Wie in Abschnitt 6.3.1.2 beschrieben ist es möglich, anhand einer einfachen Stromverbrauchsanalyse einer Schaltung für DES in CMOS-Technologie die 16 einzelnen Runden des Algorithmus sowie weitere Details des Programmablaufs zu erkennen. Besonders kritisch in Bezug auf die entweichende Seitenkanalinformation bei einer solchen Implementierung ist das Laden der Register. Werden alle Register gleichzeitig, d.h. mit derselben Taktflanke geladen, so führt dies dazu, daß sich der Stromverbrauch und die elektromagnetische Abstrahlung der einzelnen Flipflops, aus denen die Register bestehen, zu einem starken Impuls addieren, der dann bei einer Analyse deutlich sichtbar wird.

Um dies zu verhindern wurde im Rahmen dieser Diplomarbeit auf Architekturebene eine Methode entwickelt, bei der die in den Registern zu speichernden Bits

nicht mehr alle gleichzeitig mit derselben Taktflanke, sondern zeitlich verschoben zu mehreren Zeitpunkten des Taktes geladen werden. Dadurch findet eine zeitliche *Verschränkung* (engl. interleaving) beim Laden der Bits in die Register bzw. beim Register-Transfer statt. Die Ladezeitpunkte der einzelnen Bits sind dabei abhängig vom Zufall. Zusätzlich werden auf der Taktleitung Verzögerungselemente eingesetzt um die Verbreitung der Taktflanken und damit das Schalten der Flipflops weiter zu verzögern.

Auf diese Weise findet eine Desynchronisation bei der Abstrahlung der Register statt, da das Schalten der einzelnen Flipflops und die damit verbundene Erzeugung eines elektromagnetischen Impulses nicht mehr gleichzeitig sondern zeitlich versetzt stattfinden. Dadurch addieren sich die einzelnen (kleinen) Impulse der Registerzellen nicht mehr zu deutlich stärkeren Impulsen, die ein Angreifer bei einer Messung der Abstrahlung nutzen kann um daraus das Hamminggewicht einzelner Bits in den Registern und damit den Wert des geheimen Schlüssels zu rekonstruieren.

Die technische Umsetzung dieser Methode des *Register-Transfer-Random-Interleavings* in einer Schaltung wird in Kapitel 7.3.2 genauer beschrieben. Da die Maßnahme auf Architekturebene ansetzt kann sie für unterschiedliche Algorithmen verwendet werden.

6.4.5 Softwareebene

Bei einer Implementierung in Software besteht die Seitenkanalinformation beispielsweise in der Ausführungszeit von Berechnungen. Auch hier kann durch Einfügen entsprechender Maßnahmen in das Programm oder durch Umschreiben des Algorithmus eine Dekorrelation zwischen dieser Information und den berechneten Daten erreicht werden. Dies kann z.B. durch das Einfügen von Dummy-Instruktionen erfolgen, die zu einer konstanten Berechnungsdauer führen, allerdings nicht zum Ergebnis beitragen oder in einem späteren Schritt wieder rückgängig gemacht werden. Beispielsweise kann der in Kapitel 6.3.1.2 beschriebene Square-and-Multiply-Algorithmus umgeschrieben werden, sodaß dessen Ausführung unabhängig vom Hamminggewicht des Exponenten und somit des geheimen Schlüssels ist.

Außerdem können zufällige NOPs (engl. no operations) oder Schleifen mit „Scheininstruktionen“ in das Programm eingefügt werden, die die Programmausführung und damit auch die Seitenkanalinformation verändern und zu einer Desynchronisation wiederholter Seitenkanalmessungen führen.

Alle auf der Verwendung von Zufall basierenden Maßnahmen können jedoch bei einer ausreichend großen Anzahl Messungen durch einen Angriff mit differentieller Analyse und Mittelwertbildung über alle Messungen entfernt werden. Sie sind daher nur in Verbindung mit anderen Gegenmaßnahmen ausreichend sicher.

Eine weitere Methode um Softwareimplementierungen sicher gegen Seitenkanalangriffe zu machen, ist die *Maskierung* der verarbeiteten Daten [Lieb05]. Ziel ist es dabei, die Eingangsdaten eines kryptographischen Algorithmus, die einem Angreifer möglicherweise bekannt sind, durch eine Maske zu verändern, sodaß während der gesamten Berechnung mit maskierten, d.h. modifizierten Daten gerechnet wird. Dadurch sind auch alle auftretenden Zwischenergebnisse verändert.

Ist der Stromverbrauch während der Ausführung abhängig von den verarbeiteten Daten und sind die Eingangsdaten einem Angreifer bekannt, so kann diese Tatsache

dann nicht mehr für einen Angriff ausgenutzt werden, da alle während der Berechnung auftretenden und für einen Angriff verwendeten Zwischenergebnisse mit einer (geheimen) Maske verknüpft wurden. Am Ende der Berechnung wird die Maske vom Ergebnis wieder entfernt und man erhält das gleiche Ergebnis wie im Fall ohne Maske. Das Prinzip der Maskierung ist Abb. 6.9 zu entnehmen.

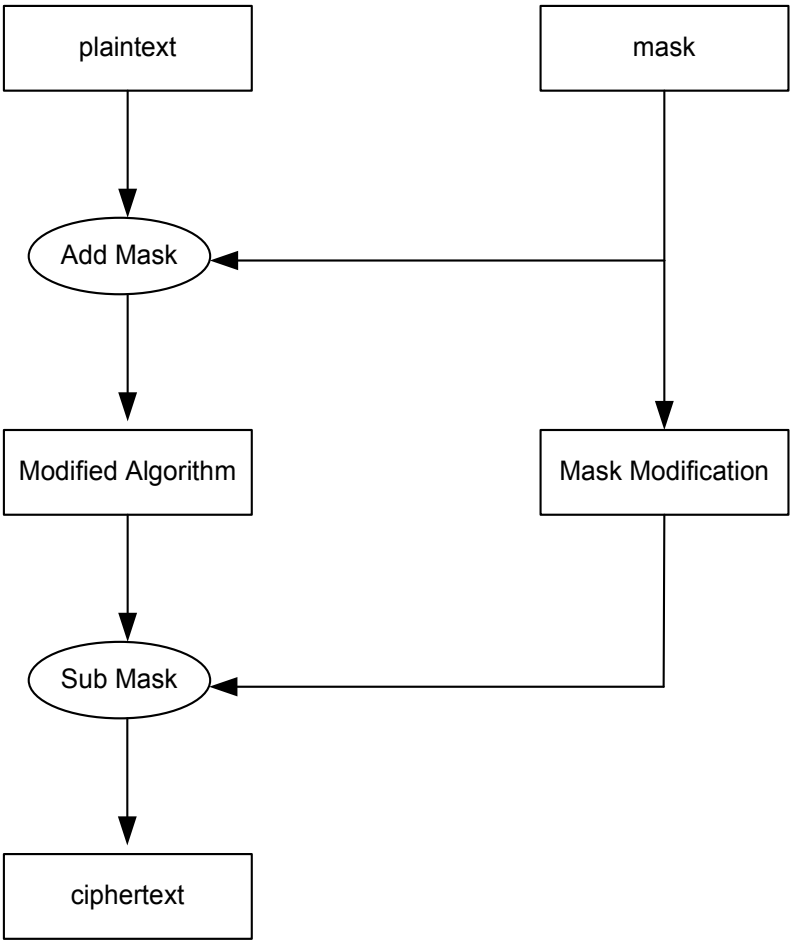

Abbildung 6.9: Prinzip der Maskierung

In den meisten Fällen wird zur Maskierung eine additive Maske verwendet, d.h. die Maskierung der Eingangsdaten erfolgt durch eine bitweise XOR-Verknüpfung mit der Maske. Am Ende der Berechnung wird die Maske dann ebenfalls durch XOR wieder vom Ergebnis abgezogen, sodaß man dasselbe Ergebnis erhält wie im Fall ohne Maske.

Eine Herausforderung bei dieser Form der Maskierung stellen die nichtlinearen Funktionen des verwendeten Algorithmus dar, da es hier sehr schwierig ist, das Ergebnis am Ende wieder zu demaskieren.

Für eine lineare Funktion f von einer Nachricht a und einer Maske m gilt

$$f(a \oplus m) = f(a) \oplus f(m).$$

Um am Ende der Berechnung dasselbe Ergebnis $f(a)$ zu erhalten wie ohne Maskierung, kann man in diesem Fall vom maskierten Ergebnis $f(a \oplus m)$ den zur Maske gehörenden Teil $f(m)$ abziehen, d.h. $f(a) = f(a \oplus m) - f(m)$.

Für die nichtlinearen Teile gilt dies nicht. Hier ist obige Gleichung nicht erfüllt, da für nichtlineare Funktionen g im Allgemeinen gilt:

$$g(a \oplus m) \neq g(a) \oplus g(m).$$

Um also am Ende dasselbe Ergebnis $g(a)$ zu erhalten wie im Fall ohne Maske, genügt es hier nicht, den zur Maske gehörenden Teil $g(m)$ zu berechnen und ihn vom erhaltenen Ergebnis $g(a \oplus m)$ abzuziehen.

Die in Blockchiffren verwendeten nichtlinearen Funktionen werden *S-Boxen* genannt. Sie werden meistens durch Lookup-Tabellen realisiert und im nichtflüchtigen Speicher des Gerätes gespeichert. Die Demaskierung der nichtlinearen Teile erfordert das Neuberechnen und Speichern dieser S-Boxen für jede Maske. Dies führt zu zusätzlichem Rechenaufwand und Speicherbedarf. Außerdem wird ein Zufallszahlengenerator benötigt um immer neue zufällige Masken zu generieren. Alternativ kann auch für jeden Durchgang dieselbe Maske verwendet werden. Diese geheime Maske kann dann jedoch, ebenso wie der geheime Schlüssel, durch einen DPA-Angriff geknackt werden.

Ein weiterer Nachteil der Maskierung ist die Tatsache, daß die für unterschiedliche Gruppenoperationen benötigten Masken ineinander umgewandelt werden müssen ohne dabei die unmaskierten Daten zu enthüllen. Im Fall des Advanced Encryption Standard (AES), dem Nachfolger von DES, kann z.B. für die Maskierung der nichtlinearen Funktionen eine multiplikative Maske anstelle der additiven verwendet werden. Diese muß jedoch vor dem nichtlinearen Teil aus der additiven gewonnen und hinterher wieder in diese zurücktransformiert werden. Diese Transformationen erfordern zusätzliche Operationen und erhöhen den Rechenaufwand.

Multiplikative Maskierungen sind außerdem mit der sogenannten *Zero-Value-Attacke* angreifbar. Treten während der Berechnung als Zwischenergebnisse in einer Runde Null-Bytes (engl. all-zero bytes) auf, so werden diese durch die multiplikative Maskierung $a \otimes m$ nicht effektiv maskiert. Sie bleiben unverändert, da $0 \otimes m = 0$. Dadurch gibt es also Stellen während der Berechnung, in denen unmaskierte Werte auftreten und die somit eine Angriffsmöglichkeit für Seitenkanalangriffe bieten.

Insgesamt wird bei der Maskierung kein datenunabhängiger Stromverbrauch, d.h. keine Dekorrelation zwischen Stromverbrauch und verarbeiteten Daten erreicht. Allerdings wurden die Daten, von denen der Stromverbrauch abhängt, durch die Maske verändert. Dadurch gibt es zwar einen Zusammenhang zwischen Stromverbrauch und diesen (maskierten) Daten, jedoch nicht zwischen Stromverbrauch und den tatsächlichen (unmaskierten) Daten. Da ein Angreifer die Maske nicht kennt, ist es ihm auch nicht möglich, die Korrelation zwischen Stromverbrauch und den maskierten Daten auszunutzen und auf diese Weise den geheimen Schlüssel zu enthüllen.

7. Entwurf und Realisierung

Im praktischen Teil dieser Diplomarbeit wurde der Data Encryption Standard (DES) als Schaltung in Hardware realisiert und mit dieser ein FPGA[1] programmiert. Die Schaltung kann durch einfache Modifikation zu einem Triple-DES ergänzt werden, wie er von der europäischen Verordnung für die digitalen Tachographen zur sicheren Kommunikation zwischen Fahrzeugeinheit und Bewegungssensor gefordert wird.

Ziel der Entwicklung war es, die Schaltung ausreichend sicher gegen (Seitenkanal-) Angriffe zu machen, um die in der Verordnung verlangten Sicherheitsanforderungen zu erfüllen und die entsprechende Sicherheitsstufe (EAL 4+) zu erreichen.

Dazu wurde zuerst eine „normale“, d.h. ungeschützte Version der Schaltung gebaut. Anhand von Seitenkanalmessungen wurde die Anfälligkeit dieser ungeschützten Schaltung für Angriffe durch Analyse des Stromverbrauchs oder der elektromagnetischen Abstrahlung gezeigt. Später wurde auf Architekturebene eine Methode entwickelt um die Schaltung resistent gegen diese Angriffe zu machen. Im Folgenden wird die systematische Entwicklung der Schaltung sowie der eingebauten Gegenmaßnahme beschrieben.

7.1 Ausgangspunkt für die Entwicklung der Schaltung

Ausgangspunkt für die Entwicklung einer Schaltung für DES war die schematische Darstellung in Abb. 7.1. Sie zeigt die Struktur einer synchronen Schaltung für DES, die von Ke Chen in [Chen05] realisiert wurde.

Daß die Schaltung „synchron“ ist bedeutet in diesem Fall, daß die Ausführung durch einen Takt gesteuert wird. Jede der 16 Runden von DES wird in genau einem Takt ausgeführt, sodaß nach 16 Takten das Ergebnis der Berechnung zur Verfügung steht. Der Takt läuft dabei kontinuierlich und soll erst auf ein Startsignal hin zur Schaltung durchgelassen werden und die Ausführung starten. Nach den 16 Runden soll die Schaltung auf ein Stopsignal hin keinen weiteren Takt erhalten und damit eine weitere Ausführung verhindert werden.

[1] Field Programmable Gate Array

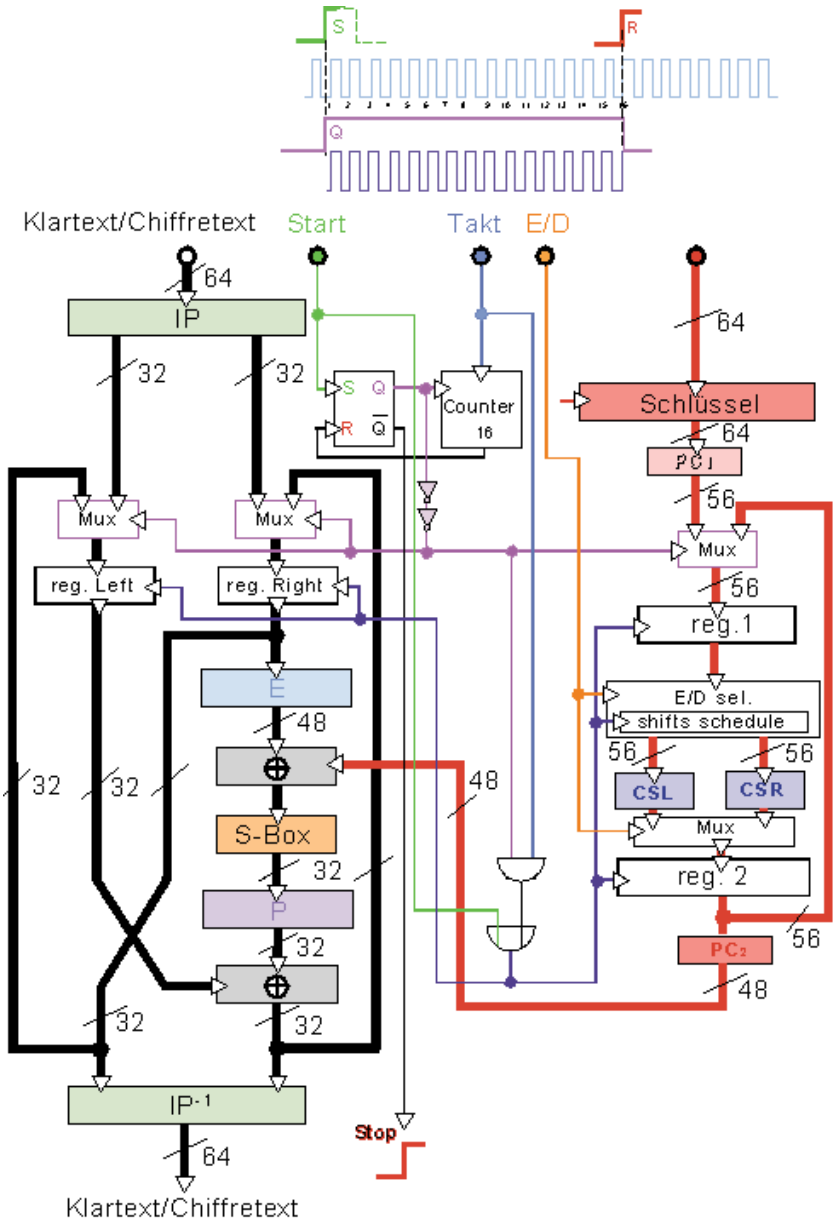

Abbildung 7.1: Schematische Darstellung einer synchronen Schaltung für DES

Der rechte Teil der Schaltung ist der *Schlüsselverarbeitungsteil* (engl. key schedule block), in dem für jede Runde ein Rundenschlüssel erzeugt wird. Im linken Teil, dem *Datenverarbeitungsteil* (engl. data path block), werden die Daten (Klartexte oder Chiffrate) verarbeitet und die Rundenfunktionen des DES ausgeführt, die auch die Verknüpfung mit dem Rundenschlüssel beinhaltet.

Weiterhin gibt es eine Leitung, über die ausgewählt werden kann, ob eine Ver- oder Entschlüsselung durchgeführt werden soll. Je nachdem müssen die Schlüsselbits bei der Generierung der Rundenschlüssel nach links oder rechts verschoben werden. In Abb. 7.1 wird dies durch die Steuerleitung *E/D* (engl. Encode/Decode) realisiert.

Die einzelnen Operationen der Rundenfunktion von DES (Expansion der rechten Seite, XOR-Verknüpfung mit dem Rundenschlüssel, Substitution durch die S-Boxen, P-Permutation und XOR-Verknüpfung mit der linken Seite) müssen bei einer synchronen Ausführung nur einmal in der Schaltung realisiert werden. Sie werden dann

in jeder Runde mit anderen Werten ausgeführt. Dies entspricht der Verwendung einer Schleife bei einer Realisierung in Software und spart außerdem Fläche auf einem Chip. Die Permutationen IP (Initiale Permutation), E (Expansion), P (P-Permutation), IP^{-1} (Finale Permutation) sowie die beiden Schlüsselpermutationen PC1 und PC2 (Permuted Choice 1 und 2) werden in der Schaltung als reine Verdrahtungen realisiert. Diese funktionieren asynchron, d.h. taktunabhängig. Die XOR-Operationen werden aus Standard-Logikgattern aufgebaut.

Für die Realisierung der S-Boxen in der Schaltung gibt es verschiedene Möglichkeiten [Chen05]. Eine davon, die in dieser Arbeit verwendet wurde, ist die Realisierung als Lookup-Tabellen, die jeden der $2^6 = 64$ möglichen Eingangswerte auf einen Ausgangswert abbilden.

In der Schaltung in Abb. 7.1 wurden im Datenverarbeitungsteil nur zwei Register, bezeichnet mit *reg. Left* und *reg. Right*, verwendet. Sie sind jeweils 32 Bit breit und dienen der Speicherung der linken und rechten Hälfte L_i und R_i in jeder Runde. Im Schlüsselverarbeitungsteil wurden dagegen zwei 56 Bit breite Register (*reg. 1* und *reg. 2*) eingesetzt, in denen jeweils der aktuelle Rundenschlüssel vor und nach der zyklischen Verschiebung enthalten ist.

Über die Steuerleitung E/D wird ausgewählt, ob eine Verschiebung der Schlüsselbits nach links (engl. cyclic shift left, CSL) oder nach rechts (engl. cyclic shift right, CSR) durchgeführt werden soll. Außerdem unterscheidet sich die Anzahl der Bits, um die der Schlüssel pro Runde verschoben wird. Für die Auswahl zwischen Verschlüsselung und Entschlüsselung, d.h. zwischen CSL und CSR, sowie der Anzahl der Bits, um die verschoben wird, werden Multiplexer (Mux) eingesetzt.

Da das Ergebnis einer Runde jeweils die Eingabe für die nächste Runde darstellt, gibt es außerdem eine Rückkopplung, über die dieses Ergebnis in die Eingangsregister für die nächste Runde kopiert wird. Eine Ausnahme stellt hierbei die erste Runde dar, die als Eingabe das Ergebnis der initialen Permutation (IP) bzw. der Schlüsselpermutation PC1 erhält. Die Auswahl zwischen Übernahme der Werte von IP und PC1 (in der ersten Runde) oder der Werte der vorherigen Runde wird ebenfalls mit Hilfe von Multiplexern getroffen.

7.2 Ungeschützte Version der Schaltung

Die erste Modifikation der Schaltung aus Abb. 7.1 war das Einfügen zweier zusätzlicher 32-Bit-Register im Datenverarbeitungsteil der Schaltung am Ende der Runde, d.h. nach der XOR-Verknüpfung der modifizierten rechten mit der linken Seite. In diesen Registern wird das Ergebnis einer Runde zwischengespeichert, bevor es in die oberen Register als Eingangswert für die nächste Runde übernommen wird.

Abb. 7.2 zeigt schematisch den ersten Entwurf der Struktur der neuen Schaltung. Die zusätzlich eingefügten Register im Datenverarbeitungsteil der Schaltung wurden mit *D-Reg L* und *D-Reg R* bezeichnet und sind jeweils 32 Bit breit. Die Register *U-Reg C* und *U-Reg D* im Schlüsselverarbeitungsteil der Schaltung sind jeweils 28 Bit breit und entsprechen dem Register *reg. 1* in Abb. 7.1. Analog dazu entsprechen die Register *D-Reg C* und *D-Reg D* dem Register *reg. 2*. Vor den Registern *U-Reg L* und *U-Reg R* gibt es Multiplexer, über die ausgewählt werden kann, ob die Werte von oben, d.h. von der initialen Permutation IP, oder die der vorherigen Runde über

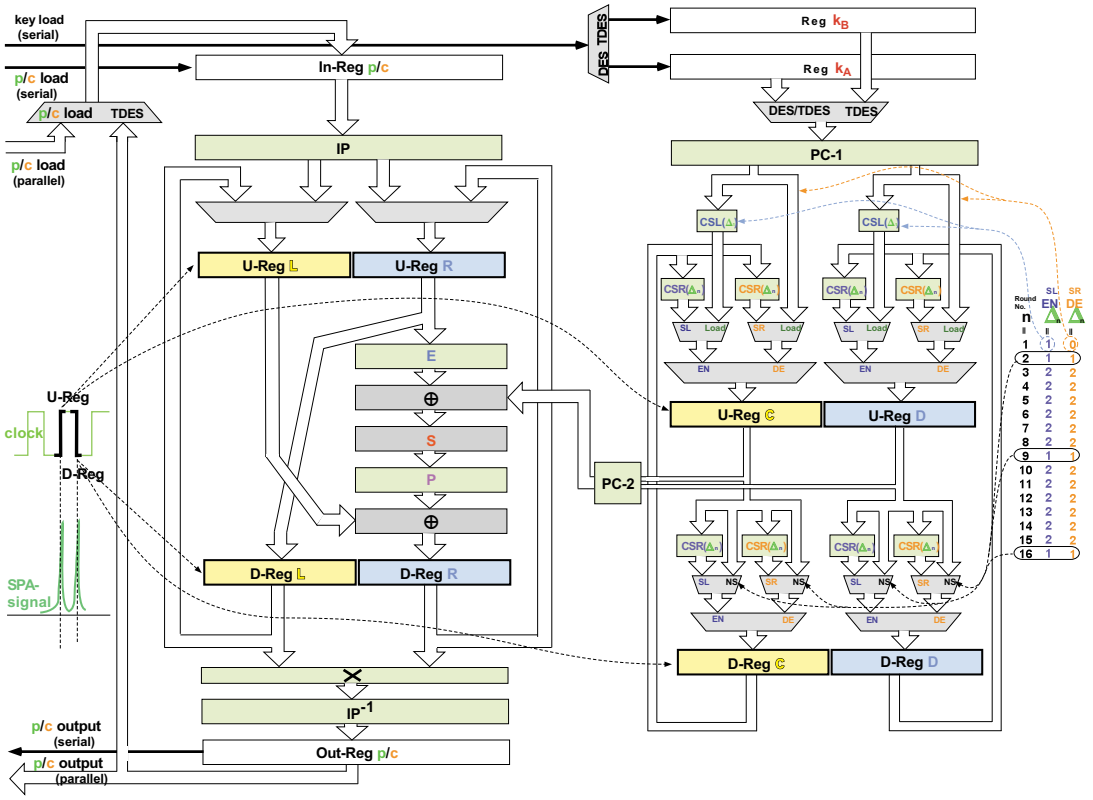

Abbildung 7.2: Schematische Darstellung der Struktur der neuen Schaltung

die Rückkopplung von den unteren Registern übernommen werden sollen. Letzteres ist in allen Runden mit Ausnahme der ersten der Fall.

Die Taktung der unteren Register (*D-Reg*) sollte genau invers (d.h. um einen halben Takt verschoben) zur Taktung der oberen Register sein. Dazu wurde der Takt, mit dem die oberen Register gesteuert werden, invertiert und zur Taktung der unteren Register verwendet. Beide Register reagieren auf steigende Taktflanken.

Die neue Schaltung sollte außer einem DES auch noch Triple-DES realisieren. Dies entspricht der dreimaligen Hintereinanderausführung von DES, bei der mindestens zwei verschiedene Schlüssel k_A und k_B verwendet werden (siehe Kapitel 5.4).

Im Schlüsselverarbeitungsteil der Schaltung in Abb. 7.2 gibt es daher zwei Register *Reg* k_B und *Reg* k_A für die beiden Schlüssel k_A und k_B. Über zusätzliche Multiplexer im Daten- und Schlüsselverarbeitungsteil kann dann ausgewählt werden, ob ein DES oder Triple-DES ausgeführt werden soll. In letzterem Fall wird das Ergebnis einer Ausführung von DES wieder als Eingabe für die nächste Ausführung verwendet. Daher gibt es eine zusätzliche Rückkopplung vom Ausgaberegister *Out-Reg* zum Eingaberegister *In-Reg*.

Wie in Abschnitt 6.3.1.2 beschrieben ist es möglich, anhand einer einfachen Stromverbrauchsanalyse einer Schaltung für DES in CMOS-Technologie die Schlüsselverschiebungen um ein oder zwei Bit zu identifizieren (siehe Abb. 6.7). Dies ist genau dann möglich, wenn diese Verschiebung durch ein Schieberegisters erfolgt, das von einem Takt gesteuert wird. Um dies zu vermeiden, wurde die Verschiebung der Schlüsselbits hier nicht durch ein Register, sondern kombinatorisch, d.h. als reine Verdrahtung realisiert.

Bei einer Verschlüsselung wird der Schlüssel je nach Runde um ein oder zwei Bit nach links verschoben. Bei einer Entschlüsselung wird er um kein, ein oder zwei Bit nach rechts verschoben. Die Anzahl der Bits, um die pro Runde verschoben wird, wird mit $\triangle n$ bezeichnet, wobei n die jeweilige Runde ist. Sie ist in der Tabelle rechts in Abb. 7.2 nochmals dargestellt.

Die Verschiebung um ein Bit (nach links oder rechts) findet jeweils am Ende einer Runde statt, wenn das Ergebnis der Runde aus den unteren Registern (*D-Reg C* und *D-Reg D*) in die oberen kopiert wird. Mit Ausnahme der ersten Runde bei der Entschlüsselung findet in jeder Runde, sowohl bei der Ver- als auch bei der Entschlüsselung, mindestens eine Schlüsselverschiebung statt. Es muß über Multiplexer lediglich ausgewählt werden, ob dies eine Verschiebung nach links oder nach rechts ist. Dies erfolgt in Abb. 7.2 über die mit *EN* und *DE* (für Encode und Decode) beschrifteten Multiplexer.

Im Fall der Entschlüsselung findet in der ersten Runde gar keine Verschiebung statt, da dort $\triangle 1 = 0$ gilt. Es wird also direkt der durch PC1 permutierte Schlüssel verwendet. Daher gibt es zusätzliche Multiplexer, die zwischen *Load* und *SR* (shift right) bzw. *SL* (shift left) auswählen. Im Fall *Load* werden die Daten von oben, d.h. von PC1 übernommen, im anderen Fall aus den unteren Registern. *Load* wird daher nur in der ersten Runde ausgewählt.

Bei der Verschlüsselung wird der Schlüssel dagegen auch in der ersten Runde um ein Bit verschoben, da hier $\triangle 1 = 1$ gilt. Es findet also auch hier ein *Cyclic Shift Left* (CSL) statt. Die zweite Verschiebung der Schlüsselbits, die bei der Ver- und Entschlüsselung jeweils in den Runden 3 bis 8 und 10 bis 15 stattfindet, wird in der Schaltung zwischen den oberen und unteren Registern ebenfalls als Verdrahtung realisiert. Es kann dort wieder über Multiplexer ausgewählt werden, ob diese zweite Verschiebung überhaupt stattfinden soll - dies ist in den eben genannten Runden der Fall - und ob nach links (bei Encode) oder nach rechts (bei Decode) verschoben werden soll.

Der Takt zur Steuerung der oberen Register wird mit *U_clk*, der für die unteren Register mit *D_clk* bezeichnet. Nachdem die an den oberen Registern (*U-Reg*) anliegenden Werte mit der steigenden Flanke von *U_clk* übernommen wurden, muß genügend Zeit vergehen, um alle Operationen der Rundenfunktion durchzuführen, bevor die Werte dann von den unteren Registern übernommen werden. Werden sie zu früh in die Register übernommen, so gibt es falsche Zwischenergebnisse und damit auch ein falsches Gesamtergebnis der Berechnung. Die Taktflanke zur Übernahme der Werte in die unteren Register (*D-Reg*) darf also erst dann kommen, wenn alle Operationen durchgeführt wurden und die Werte auf den Leitungen stabil anliegen. Da dies bei einer rein kombinatorischen Schaltung jedoch extrem schnell passiert, genügt es, die unteren Register jeweils mit der fallenden Flanke von *U_clk*, die der steigenden Flanke von *D_clk* entspricht, zu takten.

Die Steuerung der Schaltung wurde nicht von einem Mikrocontroller übernommen, sondern es wurde im nächsten Schritt der Arbeit ein *Zustandsautomat* (engl. finite state machine) entworfen, der die beiden Takte *U_clk* und *D_clk* sowie alle anderen zur Steuerung des DES notwendigen Signale generiert.

7.2.1 Entwurf der Steuerung

Um die Schaltung in Abb. 7.2 zu steuern, wird zunächst der Takt *U_clk* benötigt, mit dem die oberen Register *U-Reg L*, *U-Reg R*, *U-Reg C* und *U-Reg D* getaktet werden. Der Takt *D_clk*, der invers zu *U_clk* sein soll und mit dem die unteren Register getaktet werden, kann dann durch Einfügen eines Inverters aus *U_clk* gewonnen werden. *U_clk* und *D_clk* sollen jeweils genau 16 Takte, d.h. 16 steigende Taktflanken besitzen. Danach sollen keine weiteren Taktflanken mehr erzeugt werden.

Über die Multiplexer vor den oberen Registern im Datenverarbeitungsteil und die mit *Load* und *SR* bzw. *SL* beschrifteten im Schlüsselverarbeitungsteil wird ausgewählt, ob die Werte von oben, d.h. von IP bzw. PC1, oder von den unteren Registern, d.h. dem Ergebnis der vorherigen Runde übernommen werden sollen. Dafür wird ein Signal benötigt, mit dem in der ersten Runde die Werte vom Eingang *Load* und sonst die Werte aus den unteren Registern übernommen werden. Dieses Signal, das die Multiplexer steuert, wird mit *round* bezeichnet und ist in der ersten Runde, d.h. vor der ersten steigenden Flanke von *U_clk*, auf 0 und danach auf 1.

Schließlich wird noch ein Signal benötigt, das anzeigt, ob der Schlüssel um ein zweites Bit verschoben werden soll oder nicht. Nach der 1. Runde, die wie eben beschrieben separat mit Multiplexern realisiert wird, findet nur in der 2., 9. und 16. Runde keine zweite Verschiebung statt, da $\triangle 2 = \triangle 9 = \triangle 16 = 1$. In allen anderen Runden wird der Schlüssel jeweils um 2 Bit nach links oder rechts verschoben. Das hierfür eingesetzte Signal wird mit *ADCS* (engl. additional cyclic shift) bezeichnet. Falls in einer Runde eine zweite Verschiebung durchgeführt werden soll, so muß *ADCS* zu diesem Zeitpunkt auf 0 sein, in den anderen Fällen auf 1.

Die Entscheidung über die zweite Verschiebung wird jedoch erst in der zweiten Takthälfte jeder Runde getroffen, vor der Übernahme der Werte in die unteren Register. Der Wert von *ADCS* wird daher erst bei der steigenden Flanke von *D_clk* und nicht bei der von *U_clk* abgefragt und muß vor diesen Flanken bereits auf 0 oder 1 sein. Dabei ist darauf zu achten, daß in jedem Takt von *D_clk* der Rundenschlüssel für die jeweils **nächste** Runde erzeugt wird. *ADCS* muß also beispielsweise bei der 8. steigenden Flanke von *D_clk* auf 1 sein, damit für die Erzeugung des 9. Rundenschlüssels keine weitere Verschiebung durchgeführt wird. *ADCS* entsteht aus einer ODER-Verknüpfung der 1., 8. und 15. von einem Zähler gezählten Flanke von *D_clk*. Abb. 7.3 zeigt den gewünschten Verlauf dieser Signale.

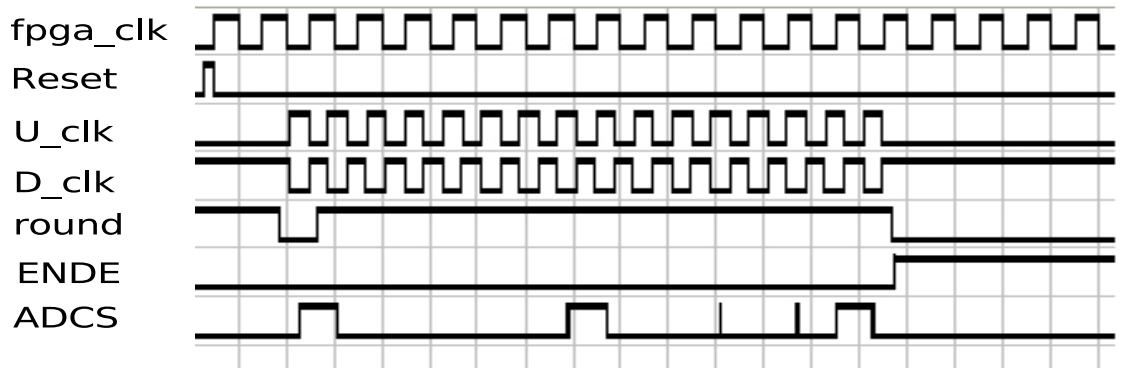

Abbildung 7.3: Verlauf der zur Steuerung der Schaltung benötigten Signale

Der Takt *fpga_clk* hat eine Frequenz von 48 MHz und wird von einem Oszillator auf dem Board, auf dem sich auch der FPGA befindet, erzeugt. Er läuft kontinuierlich so lange das Board in Betrieb ist, und wird zur Erzeugung der Takte *U_clk* und *D_clk* sowie der anderen Signale verwendet.

Das *Reset*-Signal wird vom Benutzer eingegeben und tritt daher asynchron zu *fpga_clk* zu einem unvorhersehbaren Zeitpunkt auf. Es setzt zunächst alle in der Steuereinheit verwendeten Flipflops in einen Anfangszustand zurück. Sein Inverses, das Signal *Set*, startet dann die eigentliche Ausführung. Das Signal *ENDE* schließlich zeigt das Ende der Ausführung, d.h. die Vollendung der 16. Runde an. Sobald dies auftritt, liegen vor dem Ausgangsregister *Out-Reg* (siehe Abb. 7.2) die richtigen Werte an und können in dieses übernommen werden.

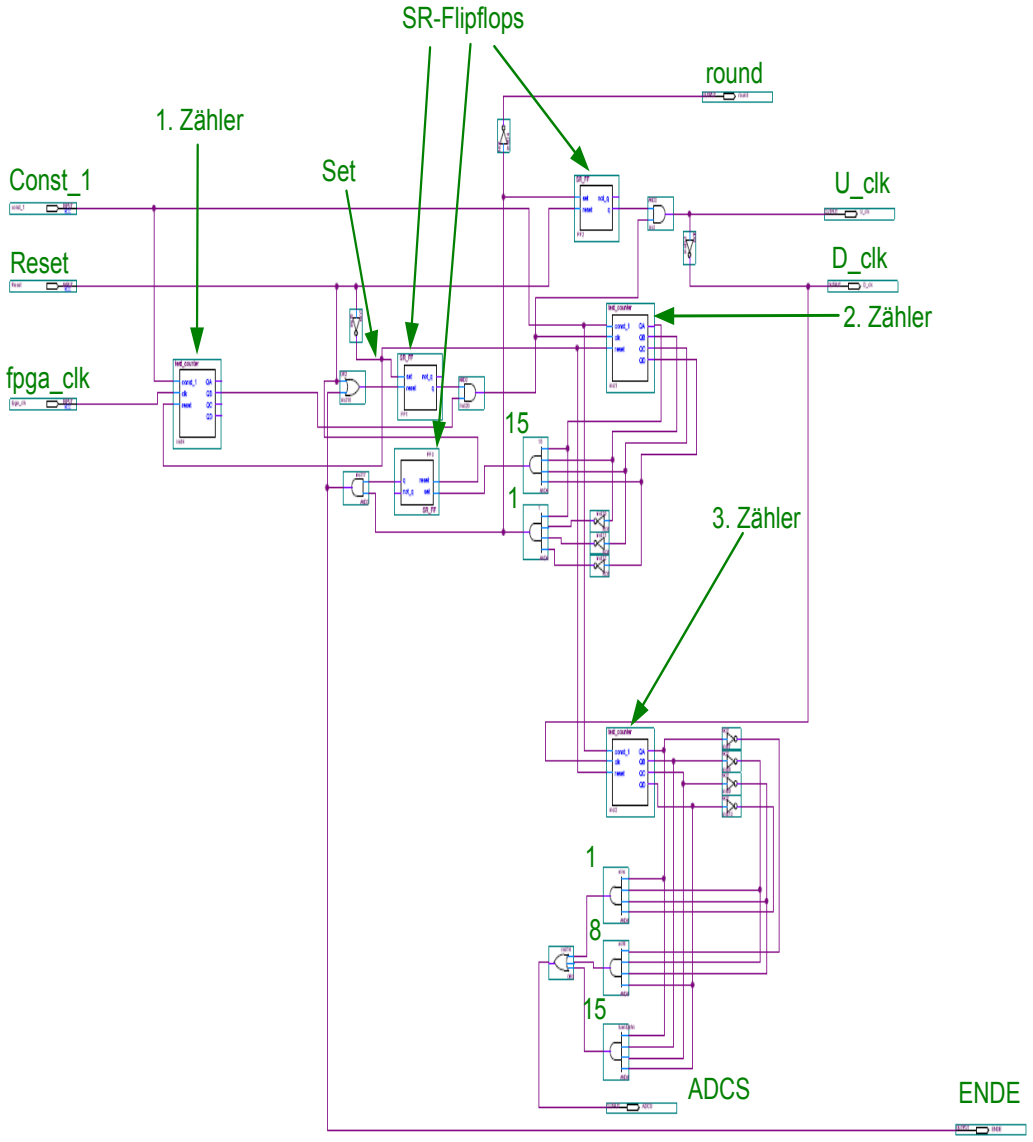

Abbildung 7.4: Mit *Quartus* entworfene Steuerungseinheit der Schaltung für DES

Abb. 7.4 zeigt die gesamte Steuereinheit der Schaltung, die die in Abb. 7.3 dargestellten Signalverläufe generiert. Die Schaltung beinhaltet drei Zähler: der erste fungiert als Taktteiler, um den (schnellen) Takt des Oszillators zu verlangsamen. Der zweite Zähler dient der Erzeugung der Signale *round*, *U_clk* und *D_clk*, wobei

letzteres durch Einfügen eines Inverters aus *U_clk* gewonnen wird. Außerdem sorgt er dafür, daß nach den 16 Takten von *U_clk* und *D_clk* keine weiteren Taktflanken mehr erzeugt werden: sobald der zweite Zähler, nachdem er bis 15 gezählt hat, wieder bei 1 beginnt, wird der Takt nicht mehr durchgelassen und es werden keine weiteren Taktflanken von *U_clk* und *D_clk* mehr erzeugt. Gleichzeitig wird dadurch das Signal *ENDE* von 0 auf 1 gesetzt. Der dritte Zähler wird zur Erzeugung des Signals *ADCS* verwendet. Da dieses erst bei einer steigenden Taktflanke von *D_clk* abgefragt wird, wird der dritte Zähler mit *D_clk* und nicht mit *U_clk* getaktet. Er arbeitet dadurch im Vergleich zum zweiten Zähler um einen halben Takt verschoben.

Die in Abb. 7.3 erkennbaren kurzen Impulse von *ADCS* zwischen der 11. und 12. sowie der 13. und 14. Taktflanke von *D_clk* rühren da her, daß es auf den Leitungen innerhalb der Zähler zu Verzögerungen der Signale kommt und dadurch die Flipflops in den Zählern nicht alle exakt gleichzeitig schalten. Dadurch kommt es kurzfristig zu falschen Werten, sogenannten *Glitches*. So tritt beispielsweise beim Übergang von $11 = 1011_2$ nach $12 = 1100_2$ kurzzeitig der Wert 1000 auf, da die Flipflops der beiden niederwertigsten Bits früher schalten als der des zweithöchsten Bits. Das Signal *ADCS* reagiert auf diesen kurzzeitigen (falschen) Wert 1000 und ist daher zu diesem Zeitpunkt ebenfalls auf 1. Die Impulse sind jedoch so kurz, daß sie die Funktionsweise der Schaltung nicht beeinträchtigen. Die weiteren Bestandteile der Schaltung in Abb. 7.4 sind Logikgatter (UND- und ODER-Gatter sowie Inverter) und Set-Reset-Flipflops.

7.2.2 Realisierung der gesamten Schaltung

Die im vorherigen Abschnitt beschriebene Steuerungseinheit (Abb. 7.4) wurde ebenso wie der Rest der Schaltung mit der Entwicklungssoftware *Quartus II Version 4.1 SP2* von *Altera* entworfen, die im Internet verfügbar ist. Dort gibt es unter anderem die Möglichkeit, schematisch Module zu erstellen, diese zu simulieren und in Verilog- oder VHDL-Code zu übersetzen. Anschließend kann mit dem Programmiergerät *ByteBlaster II* von *Altera* über die Parallelschnittstelle des PC ein FPGA programmiert werden.

Die Entwicklungsumgebung dieser Diplomarbeit ist ähnlich der in [Herm06] beschriebenen. Sie besteht aus einem FPGA-Entwicklungsboard *Digilab CC* der Firma *El-Camino*, auf dem sich ein FPGA des Typs *Cyclone EP1C12F324C7* von Altera befindet. Die Eingabe und Ausgabe von Daten wird über ein *Microcontroller Application Board CC32xxG* der Firma *Micronas* durchgeführt. Dies ist über eine Verbindungsplatine mit dem FPGA-Board verbunden. Abb. 7.5 zeigt diese Anordnung.

Das Applikationsboard wird über ein *Wiggler-Interface* der Firma *Macraigor Systems* mit der Parallelschnittstelle des PCs verbunden. Dieses Interface dient als Signalpuffer zwischen den Signalen des Parallelports des PCs und den *JTAG*[2]-Signalen des Zielgerätes.

Der Mikrocontroller wird mit der Software *Green Hills Multi2000* im Debugger-Modus betrieben. Dafür wurde ein C-Programm geschrieben, das der Kommunikation zwischen Mikrocontroller und FPGA dient. Auf diese Weise können am PC Daten (Nachrichten und Schlüssel) für den DES eingegeben und ausgewählt werden, ob eine Ver- oder Entschlüsselung durchgeführt werden soll. Die Werte werden dann

[2]Joint Test Action Group, Verfahren zum Debugging elektronischer Hardware

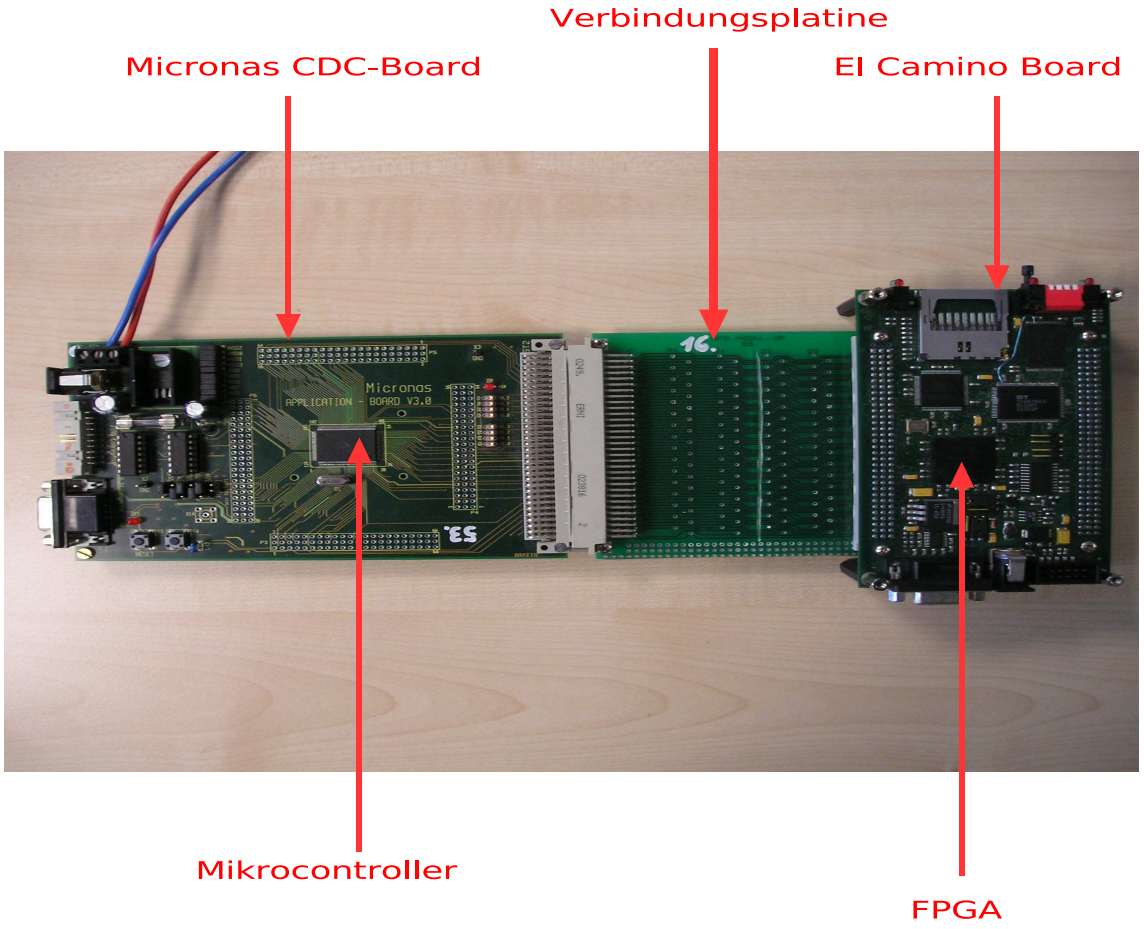

Abbildung 7.5: Entwicklungsboard mit FPGA, Mikrocontroller-Board und Verbindungsplatine

über das Applikationsboard an den FPGA geschickt. Anschließend kann ebenfalls vom PC aus das Start- bzw. Reset-Signal zum FPGA gesendet werden, das die Ausführung von DES startet. Die Daten werden dann auf dem FPGA entsprechend ver- oder entschlüsselt. Das Ergebnis der Berechnung wird dann vom Applikationsboard wieder empfangen und am PC ausgegeben.

Die Daten werden dabei seriell über eine *SPI*-Schnittstelle (engl. serial peripheral interface) an den FPGA geschickt bzw. von diesem empfangen. Auf dem Applikationsboard (im Folgenden auch als CDC-Board bezeichnet) sind zwei dieser Schnittstellen vorhanden. Die erste (*SPI0*) wird zum Senden von Nachricht und Schlüssel an den FPGA, die zweite (*SPI1*) zum Empfangen des Ergebnisses verwendet. Das Senden und Empfangen erfolgt dabei synchron, d.h. es wird durch einen eigenen Takt der *SPI*-Schnittstellen gesteuert. Dieser wird jeweils mit *SPI0_clk* bzw. *SPI1_clk* bezeichnet.

Die gesamte Schaltung des DES besteht aus drei Modulen, die jeweils einzeln erzeugt und später zusammengesetzt wurden. Das erste ist die in Kapitel 7.2.1 beschriebene und in Abb. 7.4 dargestellte Steuerungseinheit. Der zweite Teil ist der Datenverarbeitungsteil. Er entspricht dem linken Teil der schematischen Darstellung der Schaltung in Abb. 7.2. Der dritte Teil entspricht schließlich dem rechten Teil derselben Abbildung, der die Verarbeitung des Schlüssels umfasst. Die Struktur der gesamten

Schaltung ist Abb. 7.6 zu entnehmen. Zwischen dem Daten- und dem Schlüsselverarbeitungsteil verlaufen 48 Leitungen, über die der jeweilige Rundenschlüssel zur XOR-Verknüpfung mit der expandierten rechten Seite geleitet wird.

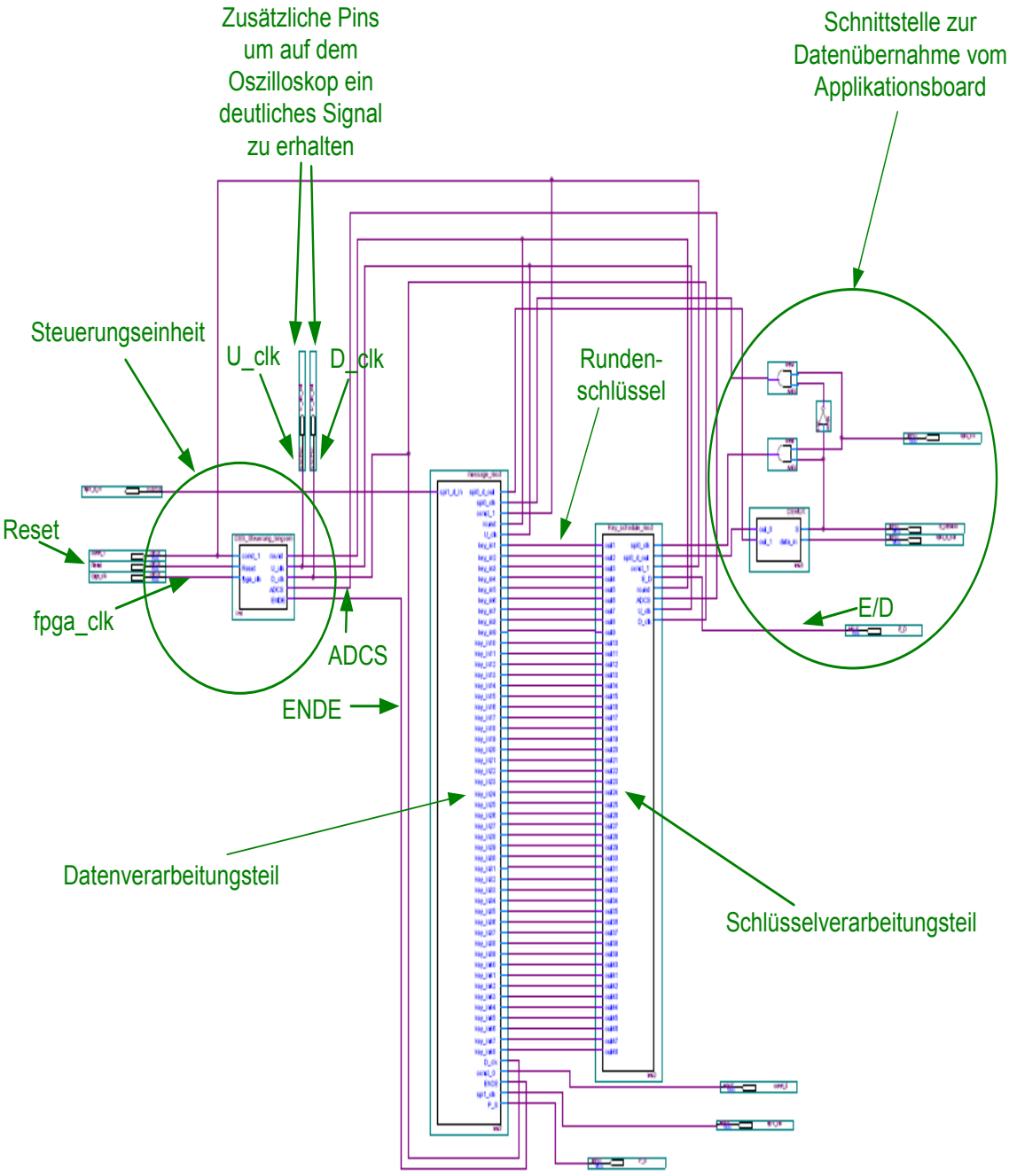

Abbildung 7.6: Schaltung für den gesamten DES

In dem in Abb. 7.6 rechts dargestellten Teil der Schaltung wird durch einen Demultiplexer (DEMUX) ausgewählt, ob über die *SPI0*-Schnittstelle des CDC-Boards gerade die Nachricht oder der Schlüssel an den FPGA gesendet wird. Je nach dem werden die Daten dann entweder zum Daten- oder zum Schlüsselverarbeitungsblock geleitet. In beiden Blöcken wird jeweils ein 64-Bit Schieberegister verwendet um die seriell ankommenden Daten in 64 Takten in den Registern zu speichern. Hierfür wird der Takt *SPI0_clk* der *SPI0*-Schnittstelle benutzt, der gleichzeitig auch zum Senden der Daten vom CDC-Board dient.

Der Schlüsselverarbeitungsteil der Schaltung ist in Abb. 7.7 im Detail dargestellt. Das 64-Bit Schieberegister, in dem die Daten vom CDC-Board ankommen, wurde mit *ShiftReg64* bezeichnet. Die Bezeichnung der anderen Register erfolgte analog zu der in Abb. 7.2 verwendeten. Sie sind jeweils 28 Bit breit. Oberhalb der Register befinden sich die Multiplexer, über die zwischen Ver- und Entschlüsselung (E/D) sowie die Runde (*round*) und die Anzahl der Bits um die verschoben wird ($ADCS$) ausgewählt werden kann. Die Permutationen PC1 und PC2 wurden wie bereits beschrieben als Verdrahtungen realisiert.

Abb. 7.8 zeigt schließlich den Datenverarbeitungsteil der Schaltung, der die Rundenfunktion enthält. Auch hier wird ein 64-Bit Schieberegister (*ShiftReg64*) verwendet um die vom CDC-Board seriell ankommenden Daten zu speichern. Ein weiteres Schieberegister (*ShiftReg64_out*) dient am Ende dazu, das Ergebnis der Berechnung seriell an das CDC-Board zurückzuschicken. Die Bezeichnung der anderen Register erfolgte in Anlehnung an Abb. 7.2. Die oberen werden mit *U_clk*, die unteren mit *D_clk* getaktet. Der Rundenschlüssel wird jeweils im Schlüsselverarbeitungsteil der Schaltung (Abb. 7.7) erzeugt, der mit dem Datenverarbeitungsteil über 48 Leitungen verbunden ist. Die übrigen Teile der Schaltung sind selbsterklärend beschriftet.

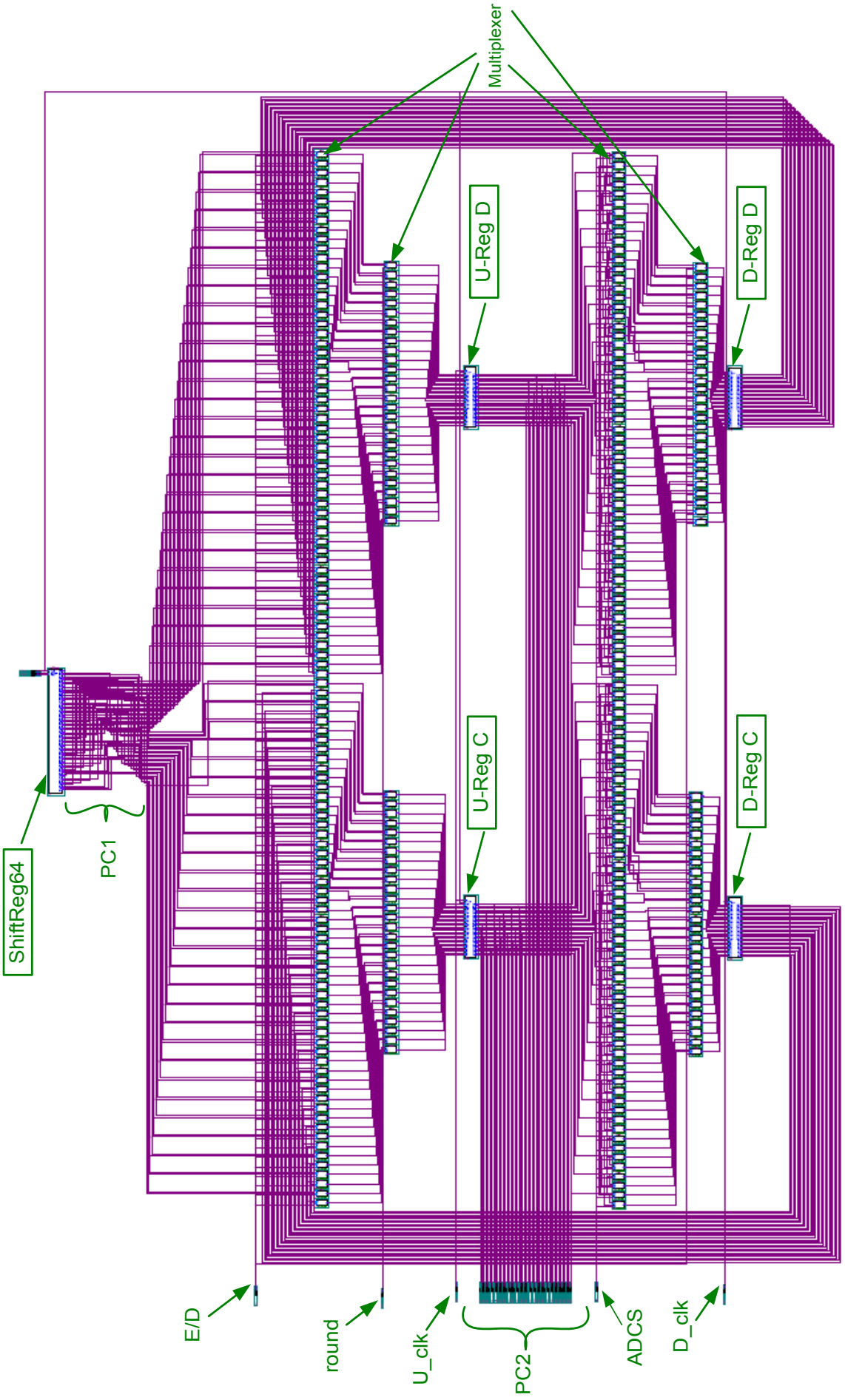

Abbildung 7.7: Schlüsselverarbeitungsteil der Schaltung

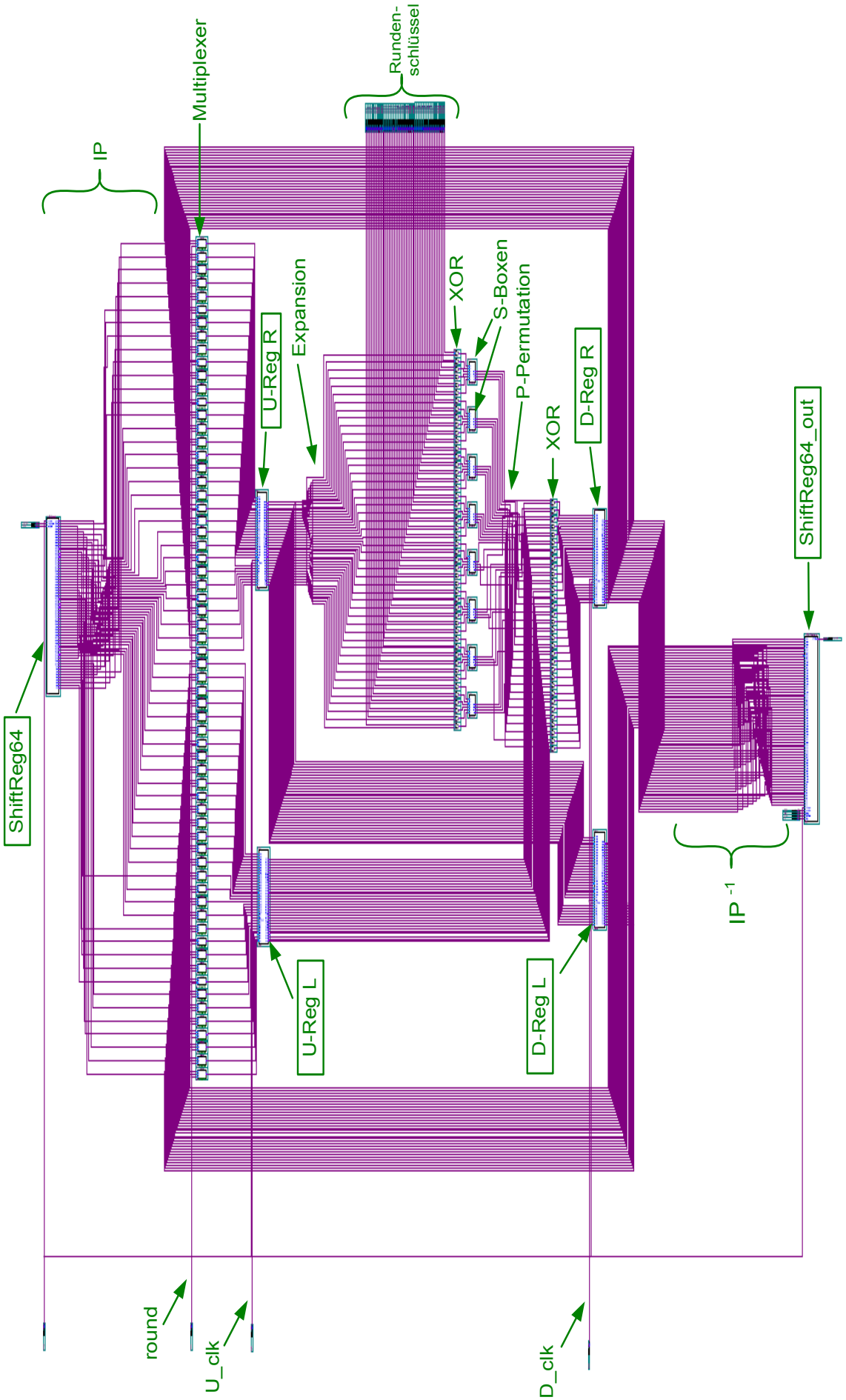

Abbildung 7.8: Datenverarbeitungsteil der Schaltung

7.2.3 Programmierung des FPGA und Messungen

Nach der Konstruktion der Schaltung wurde diese mit *Quartus* in Verilogcode übersetzt, synthetisiert und anschließend auf den FPGA geladen. Über die Verbindungsplatine zwischen FPGA- und CDC-Board waren einige Pins des Mikrocontrollers mit einigen Pins des FPGA verbunden. Diese Pins wurden mit Quartus den Ein- und Ausgängen der Schaltung zugewiesen. So wurde beispielsweise dem Eingang E/D im Schlüsselverarbeitungsteil der Schaltung, der anzeigt, ob eine Ver- oder Entschlüsselung stattfinden soll, der Pin U5.3 des Mikrocontrollers zugewiesen. Im C-Programm konnte dieser Pin dann auf 0 oder 1 gesetzt werden. Dieser Wert wurde dann zum FPGA und dort zum Eingang E/D des Schlüsselverarbeitungsteils (vgl. Abb. 7.7) geleitet. Dort wurden damit die Multiplexer gesteuert, über die bei der Generierung des Rundenschlüssels zwischen einer Verschiebung nach links (Verschlüsselung) oder nach rechts (Entschlüsselung) ausgewählt wurde.

Nachdem die Schaltung auf den FPGA geladen worden war, sollte seine elektromagnetische Abstrahlung während der Ausführung gemessen werden. Dazu wurde ein *Digital Phosphor Oscilloscope* des Typs *TDS5054B* der Firma *Tektronix* mit einer maximalen Frequenz von 500MHz und einer maximalen Abtastraste von 5GS/s verwendet.

Die zur Messung eingesetzte Sonde besteht aus einer Antenne, einem Differenzialverstärker und einem Hauptverstärker, mit dem unterschiedliche Verstärkungsfaktoren eingestellt werden können. Sie ist eine Eigenentwicklung des Europäischen Instituts für Systemsicherheit (E.I.S.S.) an der Universität Karlsruhe und wurde schon bei anderen Seitenkanalmessungen [Chen05, Herm06] verwendet. Abb. 7.9 zeigt den Versuchsaufbau für die durchgeführten Messungen.

Bei optimaler Einstellung des Oszilloskops kann man deutlich die 16 Runden von DES anhand von 16 starken elektromagnetischen Impulsen erkennen (siehe Abb. 7.10). Diese entstehen durch das gleichzeitige Laden der Register. Die Register der Schaltung bestehen aus jeweils 32 (im Datenverarbeitungsteil) bzw. 28 (im Schlüsselverarbeitungsteil) D-Flipflops, die alle an dieselbe Taktleitung angeschlossen sind. Sobald auf dieser Leitung eine positive Taktflanke kommt, schalten sie alle gleichzeitig den am Eingang anliegenden Wert auf den Ausgang durch. Alle Flipflops, deren Zustand sich dadurch ändert, d.h. die von 1 nach 0 oder von 0 nach 1 wechseln, produzieren bei diesem Umschaltvorgang einen kleinen elektromagnetischen Impuls. Da alle diese Umschaltvorgänge und damit die Abstrahlungen gleichzeitig passieren, addiert sich die elektromagnetische Abstrahlung der einzelnen Flipflops zu einem starken Impuls, der bei einer Messung mit dem Oszilloskop deutlich sichtbar wird.

Die Höhe dieses Impulses ist dabei proportional zum Hamminggewicht des Registerinhalt, falls dieses vorher leer war, bzw. zum Hamminggewicht der Änderungen des Inhalts, da nur dann ein elektromagnetischer Impuls produziert wird. Mit Hilfe statistischer Methoden und Korrelationen kann dann durch einen Angriff mit differentieller Analyse (vgl. Kap. 6.3.1.3) das Hamminggewicht des Registerinhaltes zu einem bestimmten Zeitpunkt und daraus der geheime Schlüssel rekonstruiert werden.

Bei der Messung in Abb. 7.10 wurde auf die fallende Flanke des Reset- bzw. Startsignals (gelb dargestellt) getriggert, das vom PC aus über das CDC-Board zum FPGA gesendet wird. Um ein so deutliches Signal zu erhalten war es notwendig, in der Schaltung in Abb. 7.6 zwei zusätzliche Pins einzufügen, die die Signale *U_clk*

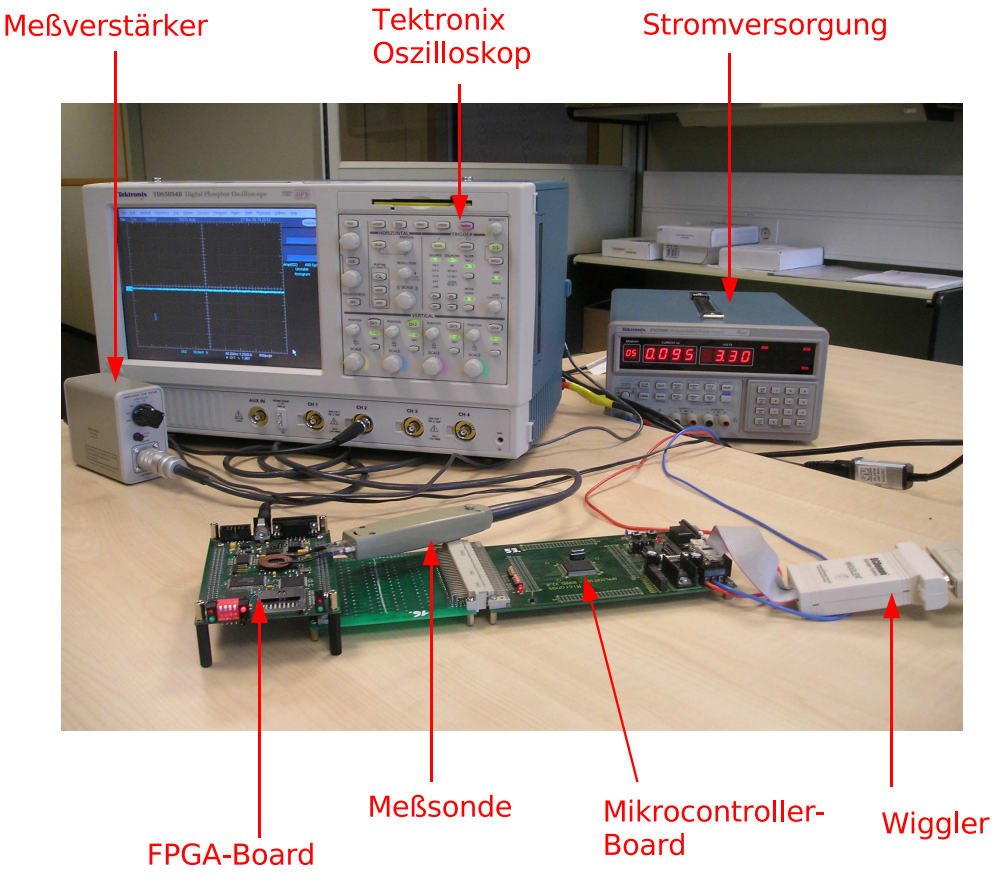

Abbildung 7.9: Entwicklungsumgebung für diese Diplomarbeit

und *D_clk* verstärken. Sie sorgen dafür, daß diese beiden Signale aus dem übrigen Rauschen hervortreten, das durch die anderen Signale auf dem FPGA verursacht wird und daß sie auf dem Oszilloskop als 16 deutliche Impulse erkennbar werden. Das Rauschen entsteht durch die Struktur eines FPGA als freiprogrammierbarer Logikschaltkreis. Im Gegensatz zu einem integrierten Schaltkreis (engl. integrated circuit, IC), der nur für eine bestimmte Anwendung entworfen und optimiert wurde und nur die Signalleitungen enthält, die in der Schaltung auch tatsächlich benötigt werden, ist ein FPGA ein reprogrammierbarer Baustein, der für viele verschiedene Anwendungen genutzt und unterschiedlich programmiert werden kann. Durch diese Universalität enthält er viel mehr (Steuer-)Leitungen als ein IC und mehr, als von der jeweiligen Schaltung tatsächlich benötigt werden. Alle diese Leitungen strahlen ein elektromagnetisches Signal ab, das zum Rauschen beiträgt und von einem Oszilloskop gemessen wird. Das Rauschen wird auch dann produziert, wenn der FPGA inaktiv ist, in diesem Fall also gerade keinen DES ausführt. In Abb. 7.10 ist dieses Rauschen vor und nach den 16 Impulsen zu erkennen. Wird die Schaltung später nicht auf einem FPGA, sondern in einem IC implementiert, so wären die 16 deutlichen Spitzen dort auch ohne die zusätzlich aktivierten Pins deutlich zu erkennen.

In [Chen05] wurde gezeigt, daß solche ungeschützten synchronen Versionen einer Schaltung für DES große Informationslecks in Form ihrer elektromagnetischen Abstrahlung besitzen und daher in Bezug auf einen Seitenkanalangriff nicht sicher sind. Durch das gleichzeitige Laden der Register in jeder Runde entstehen 16 deutlich erkennbare Impulse, die ein Angreifer bei einer Messung der elektromagnetischen

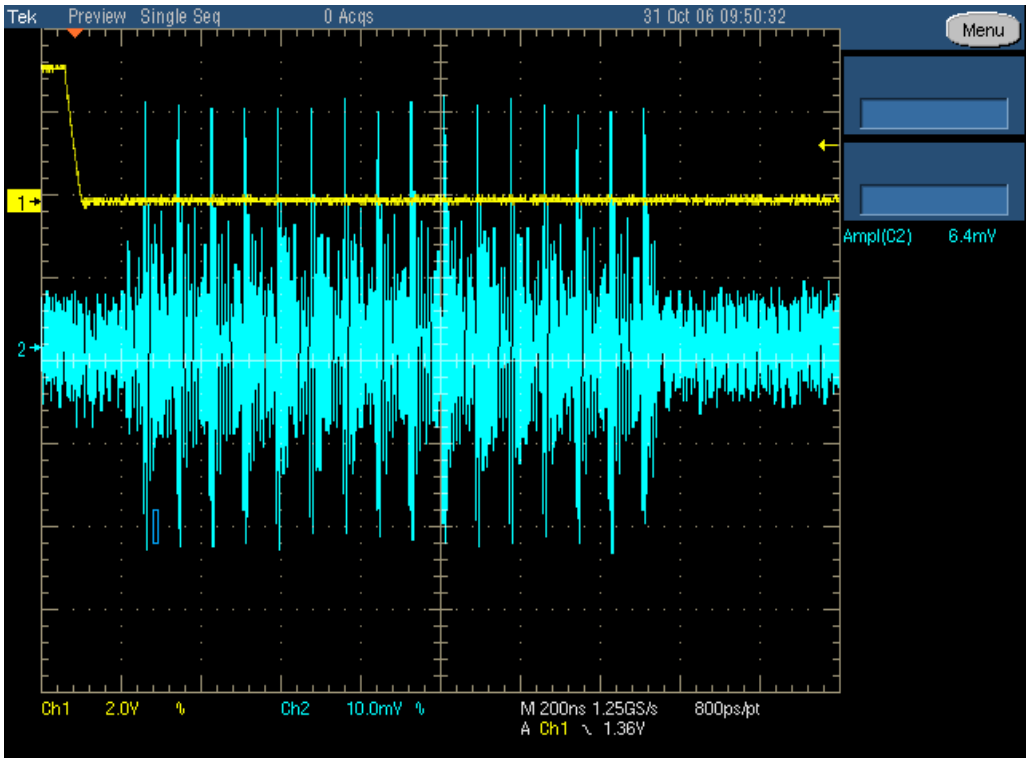

Abbildung 7.10: Messergebnis der ungeschützten Version der Schaltung für DES

Abstrahlung für einen Angriff nutzen kann, um daraus den Inhalt einzelner Register und den Wert des geheimen Schlüssels zu rekonstruieren.

Aus diesem Grund wurde im nächsten Schritt der Arbeit eine Modifikation der Schaltung vorgenommen, um eine Desynchronisation bei der Abstrahlung und damit eine Resistenz gegen solche Angriffe zu erreichen. Ziel dieser Modifikation war es, die Register nicht mehr alle gleichzeitig mit derselben Taktflanke, sondern zu verschiedenen Zeitpunkten des Taktes zu laden, sodaß die Abstrahlung der einzelnen Registerzellen zeitlich versetzt stattfindet und sich dadurch nicht mehr zu einem starken Impuls addiert.

7.3 Entwicklung einer seitenkanalresistenten Schaltung

Anhand von Seitenkanalmessungen der ersten, ungeschützten Version der Schaltung für DES konnte gezeigt werden, daß diese ein Informationsleck besitzt, aus dem Seitenkanalinformation - in diesem Fall elektromagnetische Abstrahlung - entweicht, anhand der Informationen über den geheimen Schlüssel gewonnen werden können. Besonders kritisch in Bezug auf die entweichende Seitenkanalinformation ist dabei das gleichzeitige, parallele Laden der Register. Dies führt dazu, daß sich die Abstrahlung der einzelnen Registerzellen zu den deutlich sichtbaren Impulsen in Abb. 7.10 addiert.

Um dies zu vermeiden sollte eine Methode entwickelt werden, das gleichzeitige Laden der Register zu verhindern. Es wurde dabei zunächst auf zwei Zeitpunkte innerhalb eines Taktes verteilt. Dafür waren pro Runde jeweils zwei Taktflanken von *U_clk*

bzw. *D_clk* nötig. Die Verteilung der Ladezeitpunkte der einzelnen Flipflops sollte außerdem zufällig erfolgen. Dies wurde mit Hilfe von Zufallsbits bewerkstelligt, die von einem Zufallszahlengenerator erzeugt und auf die Flipflops verteilt wurden. Je nach dem Wert dieser Bits wurden die Flipflops dann entweder mit der ersten oder der zweiten Taktflanke geladen. Abb. 7.11 zeigt die schematische Darstellung dieser modifizierten Schaltung.

Die vom Zufallszahlengenerator (engl. random number generator) *RNG* erzeugten Bits werden schematisch durch dunkle und helle Schraffierungen dargestellt. Sie repräsentieren die zufällige Verteilung der Nullen und Einsen auf die Flipflops. Über eine zusätzliche Steuerleitung (engl. register cell selector) wird dann ausgewählt, welche Flipflops mit der ersten und welche mit der zweiten Taktflanke von *U_clk* bzw. *D_clk* geladen werden.

Ziel dieses Entwurfs war es, bei einer einfachen Stromverbrauchsanalyse der Schaltung, den in Abb. 7.11 links dargestellten Signalverlauf zu erhalten, indem das Laden der Register und damit die deutlichen Impulse von einem auf zwei Zeitpunkte pro Runde verteilt wird. Dafür mußte zusätzlich zur Modifikation der Register durch das Einfügen der Zufallsbits auch eine Modifikation der Steuereinheit vorgenommen werden, sodaß diese pro Runde jeweils zwei Taktflanken von *U_clk* bzw. *D_clk* erzeugt. Der Rest der Schaltung blieb im Vergleich zur ersten Version in Abb. 7.2 unverändert.

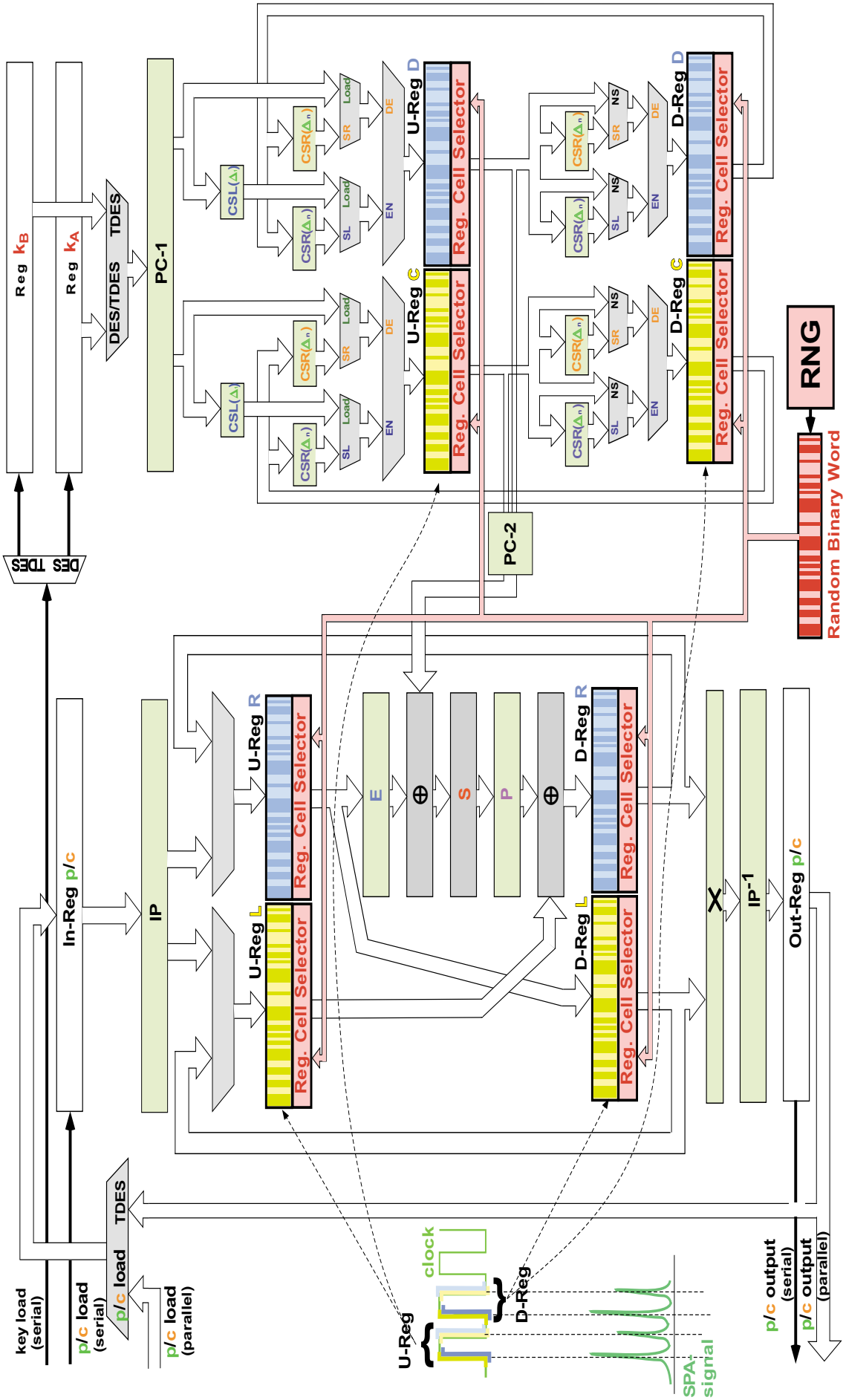

Abbildung 7.11: Schematische Darstellung der modifizierten Schaltung

7.3.1 Modifikation der Register

Um zu erreichen, daß die einzelnen Flipflops in den Registern nicht mehr alle gleichzeitig schalten, wurde zunächst vor die clk-Eingänge der Flipflops jeweils ein UND-Gatter und ein Multiplexer eingefügt. Der Ausgang des Multiplexers bildete dabei den ersten Eingang des UND-Gatters. An den zweiten Eingang wurde der Takt *U_clk* bzw. *D_clk* angeschlossen. Der Ausgang des UND-Gatters wurde dann an den clk-Eingang des Flipflops angeschlossen. Abb. 7.12 zeigt dies exemplarisch für einen Flipflop.

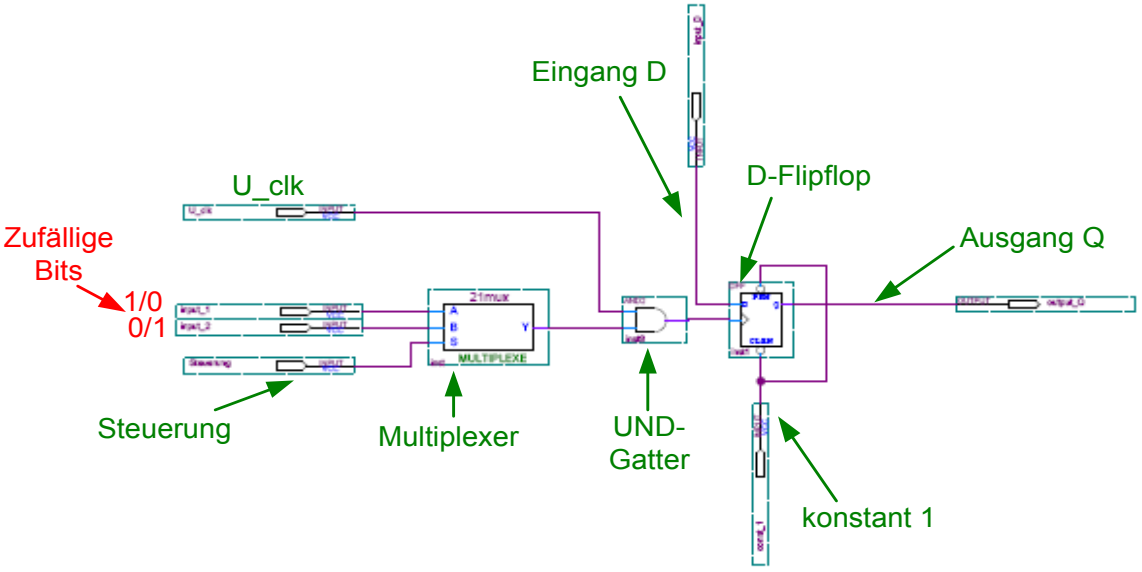

Abbildung 7.12: Modifikation der Flipflops durch Einfügen von Zufallsbits

An einem der beiden Eingänge des Multiplexers liegt jeweils eine logische 0 und am anderen eine logische 1 an. Diese werden bei den Multiplexern der einzelnen Flipflops zufällig auf die Eingänge A und B verteilt. Der D-Flipflop in Abb. 7.12 schaltet bei einer positiven Taktflanke von *U_clk* genau dann den am Eingang D anliegenden Wert auf den Ausgang Q, wenn über die Steuerleitung S des Multiplexers der Eingang ausgewählt wurde, an dem eine 1 anliegt. Diese wird dann auf den Ausgang Y und damit zum Eingang des UND-Gatters durchgeschaltet. Dort läßt sie den Takt *U_clk* zum D-Flipflop durch. Für die korrekte Funktion des Flipflops mußte an seinen Eingängen *PRN* und *CLRN* jeweils noch eine konstante 1 anliegen.

Die Multiplexer sind in Quartus so realisiert, daß im Fall $S = 0$ der Eingang B und bei $S = 1$ der Eingang A durchgeschaltet wird. Für die Register der DES-Schaltung, die aus jeweils 32 bzw. 28 der in Abb. 7.12 dargestellten (modifizierten) Flipflops bestehen, bedeutet dies: ist die Steuerleitung S der Multiplexer zum Zeitpunkt einer positiven Taktflanke von *U_clk* auf 0, so wird bei allen Multiplexern der Eingang B auf den Ausgang und damit zum UND-Gatter durchgeschaltet. Nur die Flipflops, bei denen dies eine 1 ist, bekommen einen Takt und schalten. Alle anderen, bei denen am Eingang B eine 0 anliegt, sind gesperrt. Anschließend wechselt S seinen Wert von 0 auf 1 und es wird nun der Eingang A der Multiplexer durchgeleitet. Bei der nächsten steigenden Taktflanke schalten dann wiederum nur die Flipflops, bei denen eine 1 am Eingang A und damit am UND-Gatter anliegt. Dies sind alle noch verbleibenden Flipflops, die bei der ersten Taktflanke nicht geschaltet hatten. Auf

diese Weise wird das Schalten der Flipflops und damit der gesamten Register auf zwei Zeitpunkte pro Runde verteilt.

7.3.2 Register-Transfer-Random-Interleaving

Durch das Einfügen der Zufallsbits und die Verteilung des Ladens der Register auf zwei Zeitpunkte pro Takt, findet eine zeitliche *Verschränkung* (engl. interleaving) der Bits beim Speichern in den Registern bzw. beim Transport zwischen diesen Registern statt. Da dies zusätzlich abhängig vom Zufall (engl. random) ist, wurde diese Methode als *Register-Transfer-Random-Interleaving* bezeichnet.

Dabei wird die Menge m der in den Registern zu speichernden Bits[3] in zwei Teile m_1 und m_2 aufgeteilt, sodaß gilt $m = m_1 \,|||\, m_2$. Dabei enthält m_1 die Bits, die mit der ersten Taktflanke von *U_clk* bzw. *D_clk* in die Register übernommen werden und m_2 entsprechend die restlichen Bits, die mit der zweiten Taktflanke gespeichert werden. Die Verschränkung wird mit dem Symbol ||| bezeichnet. Die Aufteilung der Bits auf die beiden Mengen findet dabei zufällig statt, wobei m_2 allerdings durch die Wahl von m_1 determiniert ist, da sie den Rest, d.h. alle übriggebliebenen Bits enthält. Die Aufteilung der Bits auf m_1 und m_2 ändert sich durch den Zufallszahlengenerator bei jeder Durchführung von DES.

Sobald ein Bit vom entsprechenden Flipflop eines Registers übernommen wurde, verbreitet sich dessen Wert über die nachfolgenden Leitungen und den kombinatorischen Teil der Schaltung. Es wirkt sich damit auch sofort auf die Ein- und Ausgänge der Logikgatter sowie der S-Boxen aus. Hier tritt aufgrund der Nichtlinearität der S-Boxen ein ähnliches Problem auf wie bei einer Maskierung der Daten, die in Kapitel 6.4.5 beschrieben wurde. Das Problem besteht darin, daß für eine nichtlineare Funktion f im Allgemeinen gilt:

$$f(a \oplus b) \neq f(a) \oplus f(b).$$

Werden also die beiden Teile a und b der Eingabe von der Funktion f getrennt voneinander durch $f(a)$ und $f(b)$ substituiert, so kann aus diesen beiden Werten nachträglich nicht das Ergebnis $f(a \oplus b)$ berechnet werden. Analog dazu kann im Fall einer Maskierung aus dem maskierten Ergebnis $f(a \oplus b)$ einer nichtlinearen Funktion f und dem zur Maske gehörenden Teil $f(b)$ nicht der unmaskierte Wert $f(a)$ rekonstruiert werden.

In der vorgestellten Methode des *Interleavings* der Bits beim Register-Transfer wird zunächst nur der zuerst übertragene Teil m_1 der Bits durch die S-Boxen substituiert und erst danach der andere Teil m_2. Aus diesen beiden Werten S-Box(m_1) und S-Box(m_2) kann im Nachhinein aufgrund der Nichtlinearität nicht der Wert S-Box($m_1 \,|||\, m_2$) rekonstruiert werden und es kommt somit zu einem falschen Ergebnis am Ausgang der S-Boxen.

Dieses Problem wird jedoch gelöst, da die Werte der zuerst geladenen Bits aus m_1 auch dann noch auf den Leitungen und an den Gattern der Schaltung anliegen, wenn mit der zweiten Taktflanke die anderen Flipflops schalten und die Werte der Bits in m_2 übernehmen. Auch diese breiten sich daraufhin auf den Leitungen aus

[3] bei einem 32-Bit-Register ist also $|m| = 32$

und wirken sich dann zusammen mit den anderen auf die kombinatorische Schaltung und die S-Boxen aus. Zu diesem Zeitpunkt werden also m_1 und m_2 gleichzeitig durch die S-Boxen substituiert und man erhält damit am Ausgang den richtigen Wert S-Box($m_1 ||| m_2$). Das Ergebnis einer Runde und damit auch das der S-Boxen darf daher erst dann von den unteren Registern übernommen werden, wenn sich ($m_1 ||| m_2$) gleichzeitig über die Leitungen und die kombinatorische Schaltung ausgebreitet und auf das Ergebnis ausgewirkt haben. Dies muß durch die Steuerung der Schaltung und die Frequenz der Takte *U_clk* und *D_clk* gewährleistet werden.

7.3.3 Steuerung des Interleavings

Da das Schalten der Register durch die Methode des Register-Transfer-Random-Interleavings auf zwei Zeitpunkte pro Runde verteilt werden und dadurch eine Verschränkung der Bits beim Speichern in den Registern stattfinden sollte, waren bei dieser Version der Schaltung zwei steigende Taktflanken von *U_clk* und *D_clk* pro Runde nötig. Außerdem wurde eine zusätzliche Steuerleitung benötigt um die Multiplexer, an denen die Zufallsbits anliegen, umzuschalten. Abb. 7.13 zeigt den gewünschten Signalverlauf.

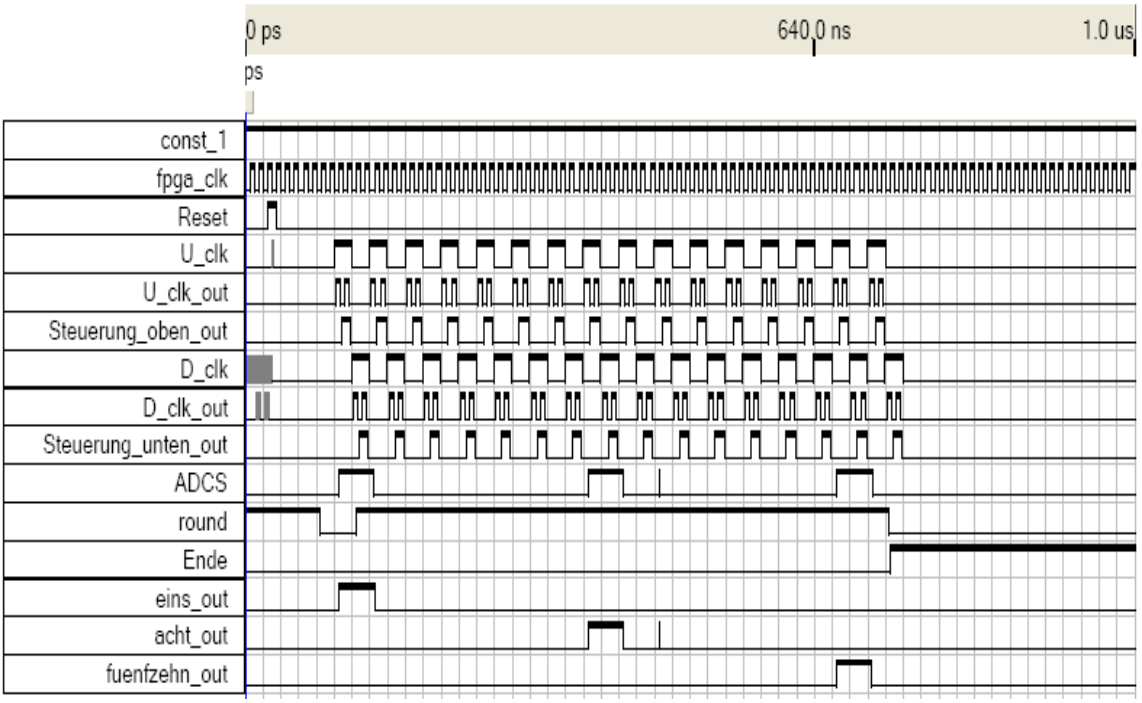

Abbildung 7.13: Signalverlauf zur Steuerung des Interleavings beim Register-Transfer

Die Takte *U_clk_out* und *D_clk_out* besitzen jeweils zwei Taktflanken pro Runde um zunächst den ersten Teil m_1 und anschließend den zweiten Teil m_2 der Flipflops zu laden. Zwischen diesen beiden Taktflanken werden die Multiplexer, an denen die Zufallsbits anliegen, durch die Signale *Steuerung_oben_out* und *Steuerung_unten_out* umgeschaltet. Diese müssen jeweils vor der ersten steigenden Flanke von *U_clk_out* bzw. *D_clk_out* auf 0 und vor der zweiten Flanke auf 1 sein. Zu Beginn des nächsten Taktes müssen sie ebenfalls wieder den Wert 0 haben. Alle anderen zur Steuerung der Schaltung verwendeten Signale sind unverändert gegenüber der ersten Version

in Abb. 7.3. Das Signal *ADCS* entsteht aus einer ODER-Verknüpfung von *eins_out*, *acht_out* und *fuenfzehn_out* wie in Kap. 7.2.1 beschrieben.

Um diesen Signalverlauf zu erhalten, wurde die in Abb. 7.14 dargestellte Konstruktion verwendet. Das Signal *U_clk_out*, das die oberen Register taktet, entsteht aus einer UND-Verknüpfung von *U_clk* und *fpga_clk*. Analog wird das Signal *D_clk_out* für die unteren Register aus *D_clk* und *fpga_clk* erzeugt. Das Signal *Steuerung_oben_out* entsteht wie abgebildet mit Hilfe eines JK-Flipflops und eines Inverters aus *U_clk_out*. Der Flipflop ändert bei jeder fallenden Flanke von *U_clk_out* seinen Wert und damit den von *Steuerung_oben_out*. Dadurch hat die Steuerleitung jeweils vor der nächsten steigenden Flanke von *U_clk_out* ihren Wert gewechselt.

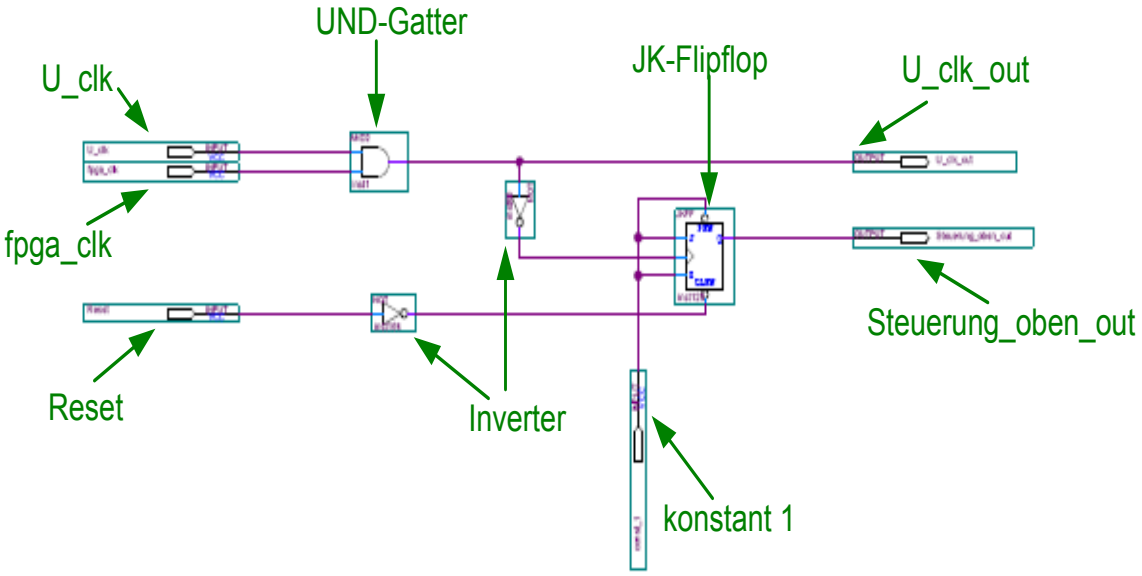

Abbildung 7.14: Erzeugung der Signale *U_clk_out* und *Steuerung_oben_out*

Für die Erzeugung des Taktes *U_clk* wird dieselbe Steuerung verwendet wie bei der ersten Version in Abb. 7.4. Für die Generierung von *D_clk* wurde allerdings eine andere Steuerungseinheit benötigt. Hätte man *D_clk* wie bei der ersten Version durch Einfügen eines Inverters aus *U_clk* erzeugt, so hätte es nicht den in Abb. 7.13 dargestellten Verlauf gehabt. Es wäre stattdessen vor der ersten steigenden und nach der letzten fallenden Taktflanke von *U_clk* jeweils gleich 1 gewesen, da *U_clk* dort gleich 0 ist. Dadurch wäre der Takt *fpga_clk* zum UND-Gatter (Abb. 7.14) durchgelassen worden und hätte damit Taktflanken von *D_clk_out* erzeugt. Dies soll jedoch nur in den 16 Runden, d.h. den 16 Takten von *D_clk* stattfinden. Vor und nach diesen 16 Takten von *D_clk* sollen keine weiteren Taktflanken von *D_clk_out* erzeugt werden.

Zur Generierung von *D_clk* wurde daher die in Abb. 7.15 dargestellte Steuerungseinheit verwendet. Ebenso wie bei der ersten Version (Abb. 7.4) wurde auch hier der durch einen Taktteiler verlangsamte Takt des Oszillators auf dem FPGA-Board zur Erzeugung von *D_clk* verwendet. Er besitzt danach nur noch ein Viertel der Frequenz von *fpga_clk* und wird daher mit *1/4 fpga_clk* bezeichnet. Die erste steigende Taktflanke von *U_clk* setzt den ersten SR-Flipflop auf 1 und ermöglicht dadurch die Erzeugung von Taktflanken von *D_clk*. Außerdem beginnt der Zähler daraufhin die Taktflanken von *D_clk* zu zählen. Zählt dieser, nachdem er bei 15 angekommen war,

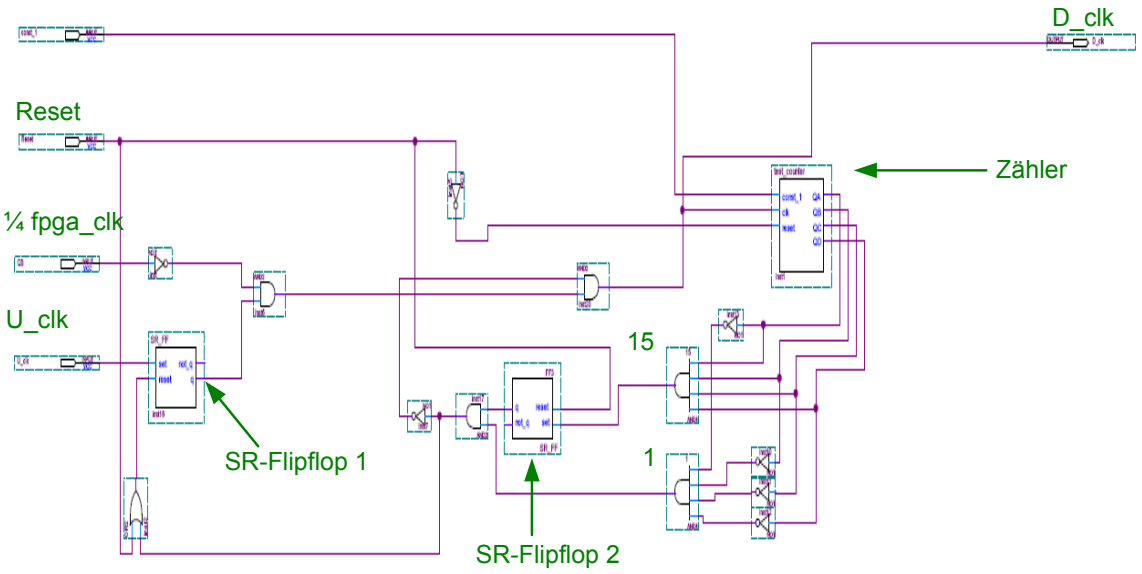

Abbildung 7.15: Erzeugung des Taktes *D_clk*

wieder die 1, so wird der 1. Flipflop zurückgesetzt und läßt damit keine weiteren Taktflanken von *1/4 fpga_clk* durch.

7.3.4 Realisierung der gesamten Schaltung mit Interleaving

Nach der Modifikation der Steuereinheit für das Register-Transfer-Random-Interleaving wurde nun die gesamte neue Schaltung zusammengesetzt (siehe Abb. 7.16). Sie besteht ebenfalls wieder aus drei Modulen, der (modifizierten) Steuereinheit, dem Daten- und dem Schlüsselverarbeitungsteil. Im Vergleich zur ersten Version enthalten diese beiden Teile nun die wie in Abb. 7.12 dargestellt modifizierten Register, die zufällig geladen werden. Die Abbildungen 7.17 und 7.18 zeigen die detaillierte Darstellung des Daten- bzw. Schlüsselverarbeitungsteils der modifizierten Schaltung.

Die Signale *U_clk_out*, *D_clk_out*, *Steuerung_oben_out* und *Steuerung_unten_out* werden im Datenverarbeitungsteil wie in Abb. 7.14 dargestellt erzeugt und dort auf Ausgänge gleichen Namens geführt. Der Schlüsselverarbeitungsteil besitzt entsprechende Eingänge[4], an denen diese Signale ankommen und dort zur Taktung der oberen und unteren Register sowie zur Steuerung der Multiplexer verwendet werden. Der Rest der Schaltung unterscheidet sich nicht von der ersten Version. Daher wurden nur die Änderungen im Vergleich zur ersten Version (Abb. 7.8 und 7.7) eingezeichnet.

[4]bezeichnet mit *U_clk_in*, *D_clk_in*, *Steuerung_oben_in* und *Steuerung_unten_in*

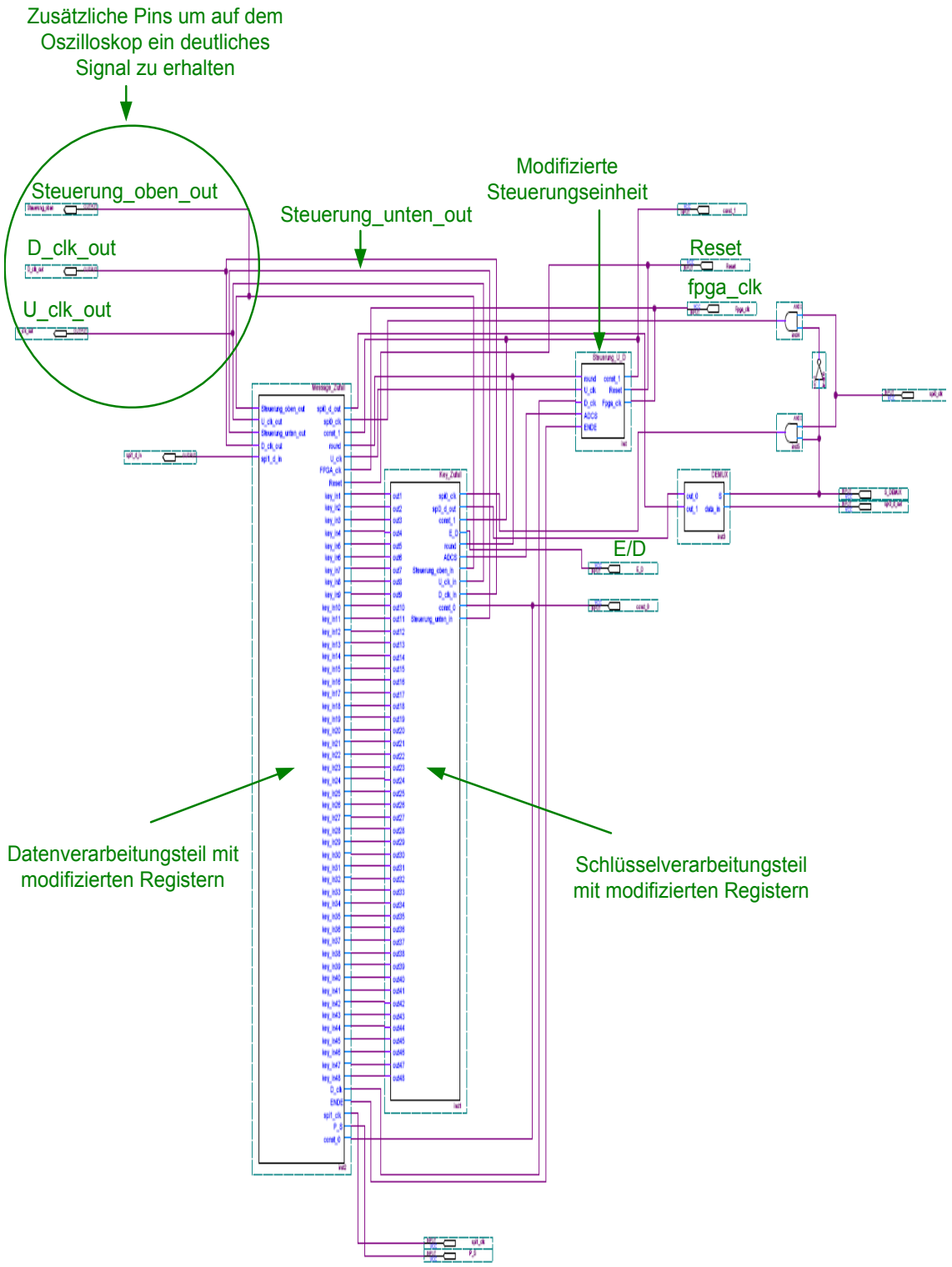

Abbildung 7.16: Modifizierte Schaltung mit Interleaving

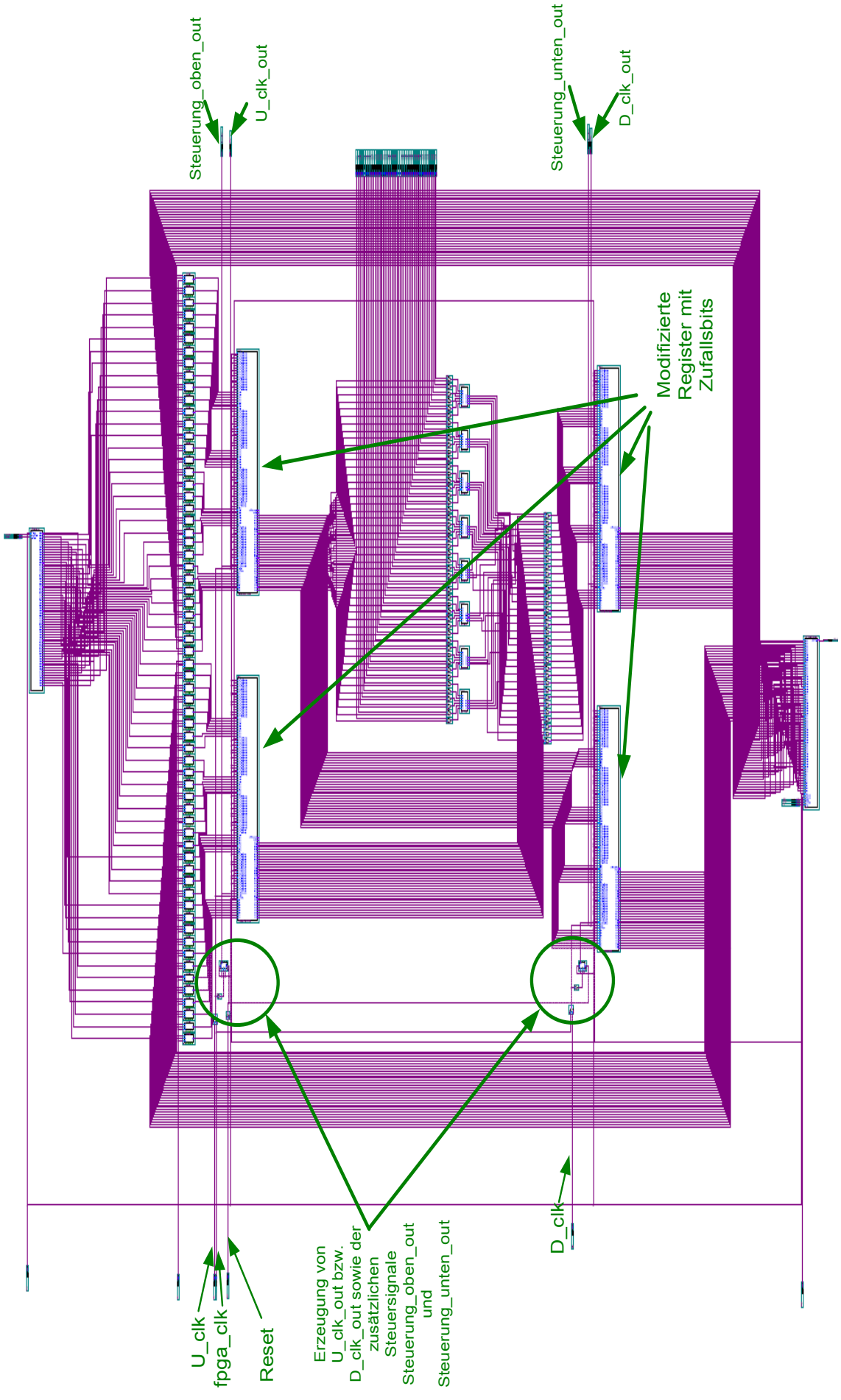

Abbildung 7.17: Modifizierter Datenverarbeitungsteil

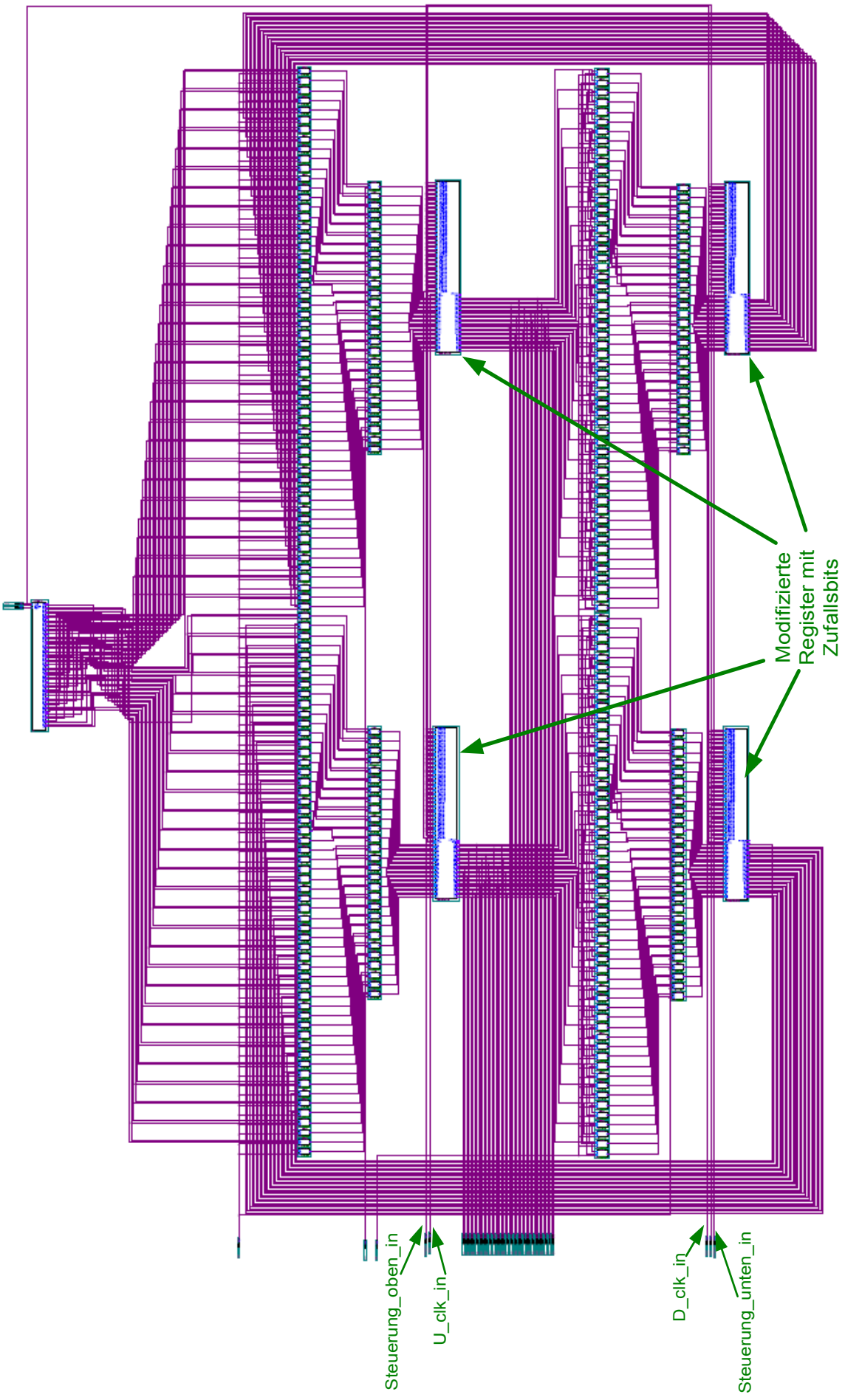

Abbildung 7.18: Modifizierter Schlüsselverarbeitungsteil

7.3.5 Messergebnisse

Mit den beschriebenen Modifikationen der Schaltung durch das Einfügen der Zufallsbits und die Erzeugung von jeweils zwei Taktflanken pro Runde, sollte ein Interleaving beim Laden der Register und dadurch eine Desynchronisation bei der Abstrahlung erreicht werden. Um dies zu überprüfen wurden erneut Messungen der elektromagnetischen Abstrahlung der Schaltung während der Durchführung gemacht. Abb. 7.19 zeigt das Ergebnis einer solchen Messung. Auch hier wurde wieder auf die fallende Flanke des Reset- bzw. Startsignals (gelb dargestellt) getriggert. Außerdem wurden vorher in der Schaltung wieder zusätzliche Pins eingefügt (siehe Abb. 7.16), um das Taktsignal zu verstärken, sodaß es aus dem übrigen Rauschen deutlich hervortritt.

Im Vergleich zu den Messungen der ersten, ungeschützten Version der Schaltung in Abb. 7.10, sind hier durch die Verwendung der zwei Taktflanken pro Runde nun anstelle von einem jeweils zwei Impulse pro Runde zu erkennen. Dabei entsteht jeder Impuls als Summe der elektromagnetischen Abstrahlungen aller Flipflops, die zu diesem Zeitpunkt ihren Zustand ändern. Die Anzahl dieser Flipflops ist dabei von der Verteilung der Zufallsbits abhängig.

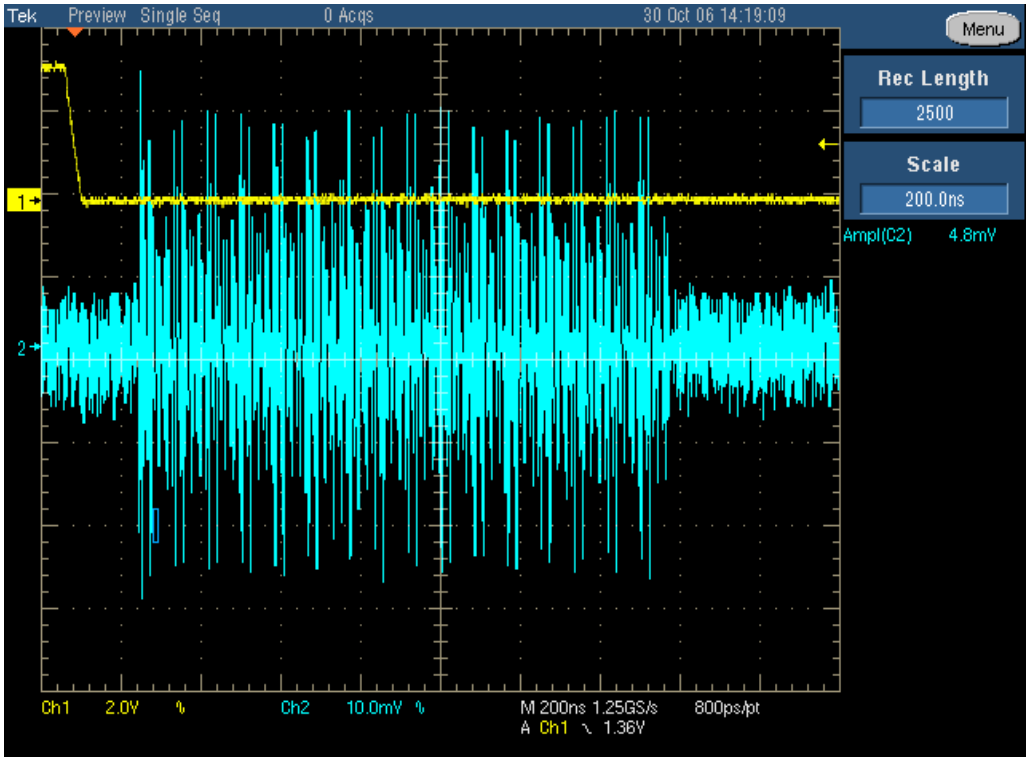

Abbildung 7.19: Messergebnis der zweiten Version der Schaltung mit Interleaving

Ein Angriff mit differentieller Analyse und Korrelationsmethoden, der sich stets nur auf ein Zwischenergebnis in einer der Runden und einen Teil des geheimen Schlüssels konzentriert, wird auf diese Weise erschwert, da nicht bekannt ist, in welchem der beiden Impulse sich das Hamminggewicht der betrachteten Bits befindet bzw. zu welchem es beigetragen hat.

Es ist dabei allerdings nicht auszuschließen, daß durch Addition der beiden Impulse einer Runde dennoch Informationen über das Hamminggewicht einzelner Bits und

den geheimen Schlüssel gewonnen werden können. Aus diesem Grund wurden im letzten Schritt der Arbeit zusätzliche Verzögerungselemente auf der Taktleitung eingefügt um das Schalten der einzelnen Registerzellen zeitlich noch mehr zu verteilen und dadurch ihre Abstrahlung weiter zu desynchronisieren.

7.3.6 Einfügen zusätzlicher Verzögerungen

In den vorherigen Abschnitten wurde das Laden der Register in der Schaltung von einem auf zwei Zeitpunkte pro Runde verteilt und außerdem mit Zufall behaftet. Dadurch kann nicht mehr rekonstruiert werden, zu welchem Zeitpunkt ein bestimmter Flipflop geschaltet und einen elektromagnetischen Impuls produziert hat. Eine Seitenkanalmessung dieser modifizierten Version der Schaltung führte zu dem in Abb. 7.19 dargestellten Ergebnis. Pro Runde sind hier zwei deutliche Impulse zu erkennen, die jeweils aus der Summe der elektromagnetischen Abstrahlung aller Flipflops entstehen, die zu diesem Zeitpunkt schalten und dabei ihren Zustand ändern.

Um das Laden der Register zeitlich noch mehr zu verteilen und damit deren elektromagnetische Abstrahlung weiter zu desynchronisieren, wurden im letzten Schritt dieser Arbeit zusätzliche Verzögerungselemente auf der Taktleitung eingefügt, um die Taktflanken von *U_clk_out* und *D_clk_out* auf dem Weg von einem Flipflop zum nächsten zusätzlich zu verzögern. Abb. 7.20 zeigt dies exemplarisch für zwei Flipflops.

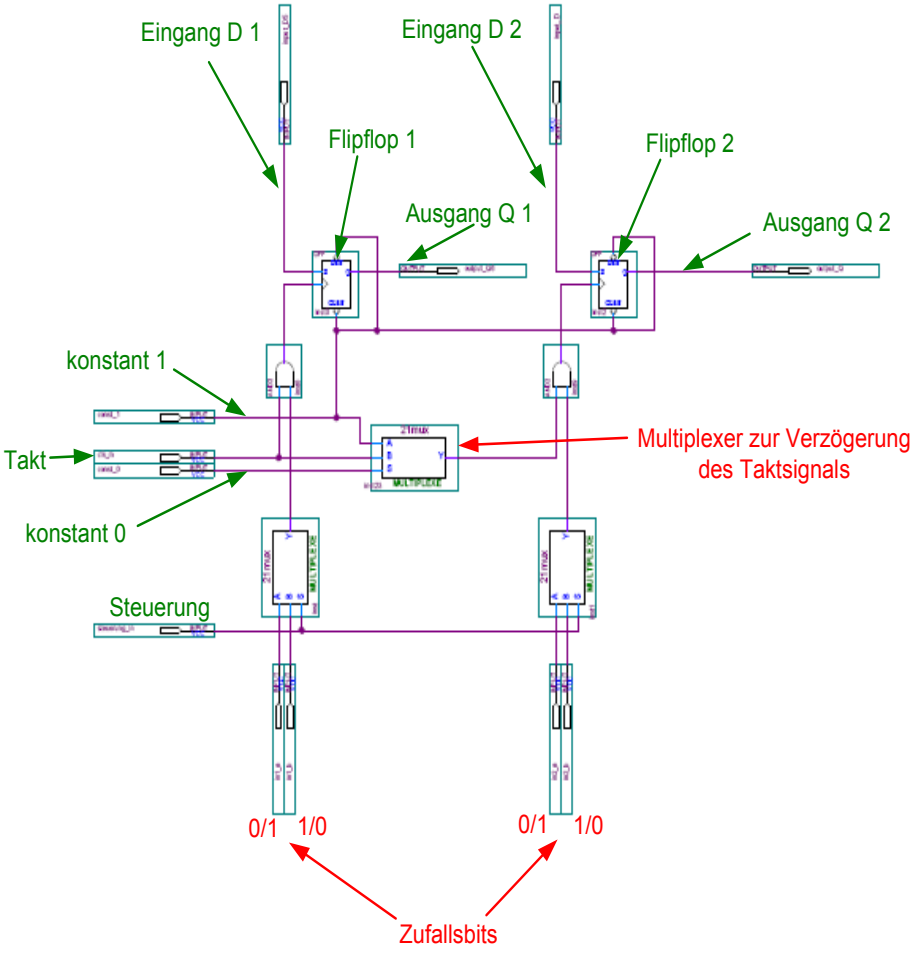

Abbildung 7.20: Einfügen zusätzlicher Verzögerungen auf der Taktleitung

Vor dem *clk*-Eingang der Flipflops befindet sich wieder jeweils ein UND-Gatter, an dessen Eingänge der Ausgang eines Multiplexers und der Takt angeschlossen sind. An den beiden Eingängen des Multiplexers liegen wieder die Zufallsbits an. Der Takt wird nur dann zum Flipflop durchgelassen, wenn über die Steuerleitung des Multiplexers der Eingang ausgewählt wurde, an dem eine 1 anliegt. In diesem Fall schalten die betreffenden Flipflops den am Eingang D anliegenden Wert auf den Ausgang Q durch. Andernfalls ist der Takt durch das UND-Gatter gesperrt und der Flipflop schaltet erst, wenn über die Steuerleitung der andere Eingang des Multiplexers ausgewählt wird.

Zwischen die clk-Eingänge der UND-Gatter vor den Flipflops wurde nun zusätzlich ein Multiplexer eingefügt, der als Verzögerungselement fungiert. Am Eingang B dieses Multiplexers ist der Takt angeschlossen. Die Steuerleitung S ist konstant auf 0, sodaß permanent der Eingang B auf den Ausgang durchgeschaltet wird. Am Eingang A liegt eine konstante 1 an, dies ist jedoch beliebig, da dieser Eingang nie auf den Ausgang geschaltet wird.

Eine am Eingang B dieses Multiplexers ankommende Taktflanke wird anschließend auf den Ausgang geschaltet und erreicht auf diese Weise das UND-Gatter des nächsten Flipflops mit einer minimalen Verzögerung gegenüber dem ersten. Der nächste Flipflop schaltet somit später als der erste und produziert dadurch - falls sich sein Zustand ändert - auch erst später einen elektromagnetischen Impuls. Ob der Flipflop bei dieser Taktflanke überhaupt schaltet oder nicht, hängt wiederum von der Verteilung der Zufallsbits und dem zweiten Eingang des UND-Gatters ab.

Durch die zusätzlichen Verzögerungen werden die Ladezeitpunkte der Flipflops zeitlich noch weiter verteilt und es kommt damit zu einer weiteren Desynchronisation der Abstrahlung der Register.

7.3.7 Messergebnisse der endgültigen Version der Schaltung

Auch von dieser endgültigen Version der Schaltung wurden anschließend Seitenkanalmessungen gemacht. Abb. 7.21 zeigt das Ergebnis einer solchen Messung. Um auf dem Oszilloskop ein deutliches Signal zu erhalten mußten auch hier wieder die Taktsignale durch zusätzliche Pins verstärkt werden. Dies führte dazu, daß auch bei dieser Version die zwei Taktflanken pro Runde anhand zweier Impulse erkennbar waren.

Durch die zusätzlich eingebauten Verzögerungen auf der Taktleitung fand das Schalten der Registerzellen nun jedoch nicht mehr nur zu diesen beiden Zeitpunkten, sondern auch zwischen und nach diesen Taktflanken statt. Der Ladezeitpunkt der Register wurde damit von zwei auf mehrere Zeitpunkte pro Takt verteilt. Dadurch fand auch deren Abstrahlung asynchron zu vielen Zeitpunkten des Taktes statt. Die Abstrahlung einer oder mehrerer einzelner Registerzellen kann daher nicht mehr bestimmt werden, da sie vom Zufall und den Verzögerungen auf der Taktleitung abhängt. Auf diese Weise ist es nicht mehr möglich, durch Addition der beiden Impulse einer Runde mit Korrelationsmethoden Informationen über das Hamminggewicht einzelner Bits oder den geheimen Schlüssel zu gewinnen, da sich diese Information nun nicht mehr nur in, sondern auch zwischen diesen Impulsen befindet.

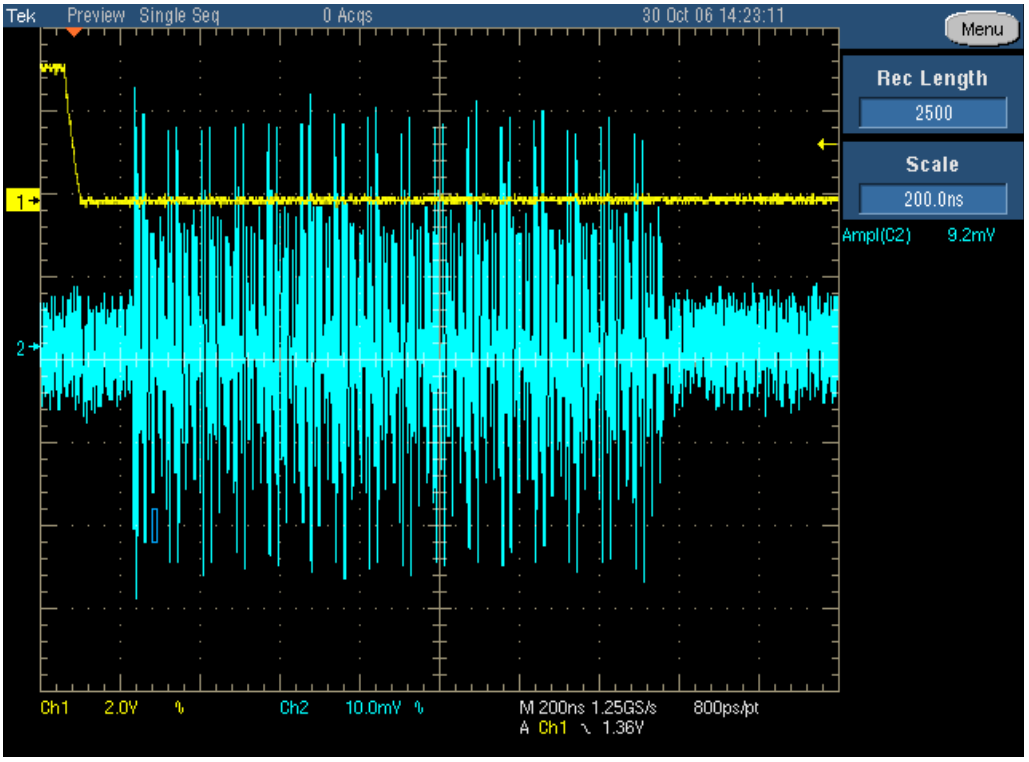

Abbildung 7.21: Messergebnis der endgültigen Version der Schaltung mit Interleaving und zusätzlichen Verzögerungen

8. Zusammenfassung und Ausblick

Eine bedeutende und zugleich mächtige Angriffsform kryptographischer Systeme sind Seitenkanalangriffe. Sie stellen eine besondere Bedrohung dar, da hierbei die mathematische Sicherheit eines Algorithmus umgangen wird. Es werden physikalische Eigenschaften der Implementierung des Algorithmus, die während der Berechnungen auftreten für einen Angriff genutzt. Auf diese Weise werden auch als sicher geltende kryptographische Algorithmen angreifbar. Die aus einem Seitenkanal entweichende Information wie z.B. der Stromverbrauch oder die elektromagnetische Abstrahlung, kann dabei genutzt werden um mit Hilfe statistischer Methoden und Korrelationen Informationen über die verarbeiteten Daten und den geheimen Schlüssel zu gewinnen.

Bei der Implementierung kryptographischer Verfahren ist es daher notwendig, entsprechende Gegenmaßnahmen gegen solche Angriffe einzubauen, sodaß die aus einem Seitenkanal entweichende Information nicht mehr für einen Angriff genutzt werden kann. Die möglichen Gegenmaßnahmen hängen dabei von der jeweiligen Implementierung und der gewählten Technologie ab. Es müssen meist mehrere Maßnahmen kombiniert werden, um einen ausreichenden Schutz und die Resistenz der Systeme gegen diese Angriffe zu erreichen.

Besonders anfällig für Seitenkanalangriffe sind eingebettete Systeme, da ein Angreifer hier im Allgemeinen leicht Zugriff auf das Gerät hat und entsprechende Seitenkanalmessungen durchführen kann. Eine mögliche Realisierungsform kryptographischer Funktionen in eingebetteten Systemen ist die Realisierung als (kombinatorische) Schaltung in Hardware. Werden dabei Register eingesetzt, so ist das Laden dieser Register in Bezug auf die entweichende Seitenkanalinformation besonders kritisch. Das Laden geschieht im Allgemeinen taktgesteuert, d.h. die am Eingang der Registerzellen anliegenden Werte werden alle gleichzeitig in die Register übernommen. Alle Registerzellen, deren Zustand sich dadurch ändert, produzieren dabei einen (kleinen) elektromagnetischen Impuls. Da die Impulse alle gleichzeitig emittiert werden, addieren sie sich zu einem großen Impuls. Die Höhe dieses Impulses bei einer Messung ist dabei proportional zum Hamminggewicht des Registerinhalts bzw. zu dessen Änderungen. Mit Hilfe statistischer Methoden und Korrelationen kann daraus

der Wert einzelner Bits und damit sukzessive auch der Wert des geheimen Schlüssels rekonstruiert werden.

Im Rahmen dieser Diplomarbeit wurde daher eine Schaltung für den Data Encryption Standard (DES) bzw. Triple-DES entwickelt, bei der das Laden der Register von einem auf mehrere Zeitpunkte pro Takt verteilt wurde. Dadurch findet die Abstrahlung der einzelnen Registerzellen nicht mehr gleichzeitig sondern asynchron statt.

Das Laden der Register wurde dabei zusätzlich mit Zufall behaftet, sodaß nicht mehr rekonstruiert werden kann, zu welchem Zeitpunkt eine bestimmte Registerzelle geladen wurde und gegebenenfalls einen elektromagnetischen Impuls produziert hat. Zusätzlich wurden auf der Taktleitung Verzögerungselemente eingesetzt um die Ladezeitpunkte der Registerzellen zeitlich noch mehr zu verteilen und dadurch deren Abstrahlung weiter zu desynchronisieren.

Die entwickelte Methode wird als *Register-Transfer-Random-Interleaving* bezeichnet, da eine zeitliche Verschränkung beim Laden der Register stattfindet, die zusätzlich mit Zufall behaftet ist. Auf diese Weise wird eine Resistenz gegen Angriffe durch Analyse der elektromagnetischen Abstrahlung erreicht. Dies wurde anhand von Seitenkanalmessungen verifiziert. Da die Maßnahme auf Architekturebene erfolgt, ist sie unabhängig vom verwendeten Algorithmus und kann für Hardware-Implementierungen verschiedener kryptographischer Verfahren eingesetzt werden.

Insgesamt gibt es jedoch keinen universellen Schutz von Implementierungen kryptographischer Verfahren vor allen bekannten Angriffen. Die Angriffe und damit auch die Schutzmaßnahmen sind abhängig von der jeweiligen Implementierungsform und der gewählten Technologie. Die Entwickler solcher Systeme müssen daher verschiedene Maßnahmen kombinieren und auf die jeweilige Implementierung anpassen um eine ausreichende Sicherheit und Resistenz gegen Angriffe zu erreichen. Die Sicherheit ist dabei jedoch nur so lange gewährleistet, wie es keine neuen und bis dato unbekannten Angriffsformen gibt.

Einsatzzweck der im Rahmen dieser Diplomarbeit entwickelten Schaltung waren die von der europäischen Union eingeführten digitalen Tachographen. Die bislang eingesetzten mechanischen Tachographen waren verhältnismäßig leicht zu manipulieren, was bei den digitalen durch den Einsatz kryptographischer Methoden verhindert werden soll. Dadurch wird die Manipulations- und Datensicherheit der Geräte verbessert und auf diese Weise auch ein Beitrag zur Erhöhung der Verkehrssicherheit geleistet.

Literaturverzeichnis

[AiOs00] M. Aigner und E. Oswald. *Softwaremodelle zur Feststellung der DPA-Anfälligkeit von Hardwaremodulen.* In *Proceedings of AUSTROCHIP 2000, Graz*, Oktober 2000.

[Ande01] R. Anderson. *Security Engineering: A Guide to Building Dependable Distributed Systems.* Wiley Computer Publishing New York. 2001.

[Blum03] Rainer Blum. *Tachofälschung. Der große Kilometerbetrug. Das Ende der Tacho-Mafia. Ein Insider packt aus.* CHARTS news Verlag. 2003.

[Chen05] Ke Chen. *Untersuchung und Realisierung einer seitenkanalresistenten Blockchiffre.* Diplomarbeit, Institut für Algorithmen und Kognitive Systeme, Fakultät für Informatik, Universität Karlsruhe (TH), 2005.

[Cryp06] CV Cryptovision. *Seitenkanalangriffe - Eine kurze Einführung.* cv cryptovision GmbH, Munscheidstr. 14, 45886 Gelsenkirchen, http://www.cryptovision.com, 2006.

[Geme02] Amtsblatt Europäische Gemeinschaften. *Verordnung (EG) Nr. 1360/2002 der Kommission vom 13. Juni 2002 zur siebten Anpassung der Verordnung (EWG) Nr. 3821/85 des Rates über das Kontrollgerät im Straßenverkehr an den technischen Fortschritt*, 2002.

[GoTy03] J. Golič und C. Tymen. *Multiplicative Masking and Power Analysis of AES.* In *Cryptographic Hardware and Embedded Systems - CHES 2002*, Band 2523 der *Lecture Notes in Computer Science*. Springer-Verlag, 2003, S. 198–212.

[Herm06] Markus Hermann. *Seitenkanalresistente skalare Multiplikation auf elliptischen Kurven.* Diplomarbeit, Institut für Algorithmen und Kognitive Systeme, Fakultät für Informatik, Universität Karlsruhe (TH), 2006.

[Inte03] International Organization for Standardization (ISO). *Road vehicles - Tachograph systems - Part 3: Motion sensor interface.* International Standard (ISO) 16844-3, 2003.

[JoKa03] J. Jonsson und B. Kaliski. *Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1*, 2003.

[KoJJ99] P. Kocher, J. Jaffe und B. Jun. *Differential Power Analysis.* In *Proceedings of Advances in Cryptology - CRYPTO 99*, Band 1666 der *Lecture Notes in Computer Science*. Springer-Verlag, 1999, S. 388–397.

[LePW06] K. Lemke, Ch. Paar und M. Wolf. *Embedded Security in Cars*. Springer-Verlag Berlin Heidelberg. 2006.

[Lieb05] Nora Lieberknecht. *Maskierungstechniken für Blockchiffren als Maßnahme gegen Seitenkanalangriffe*. Studienarbeit, Institut für Algorithmen und Kognitive Systeme, Fakultät für Informatik, Universität Karlsruhe (TH), 2005.

[LWBG01] D. Lazic, P. Wichmann, Th. Beth und W. Geiselmann. *Kryptographische Unsicherheit von Chipkarten verursacht durch kompromittierende Abstrahlung*. In *Tagungsband der fünften Chemnitz Fachtagung Mikromechanik und Mikroelektronik*, 2001, S. 259–265.

[Mang04] S. Mangard. *Securing Implementations of Block Ciphers against Side-Channel Attacks*. Dissertation, Institute for Applied Information Processing and Communications (IAKS), Graz University of Technology, 2004.

[Nati99] National Institute of Standards and Technology (NIST). *Data Encryption Standard (DES)*. Federal Information Processing Standards Publication (FIPS PUB) 46-3, 1999. Reaffirmed 1999 October 25.

[PfGr05] D. Pfeiffer und M. Grüner. *Das Digitale Kontrollgerät im Hinblick auf die aktuellen gesetzlichen und technischen Rahmenbedingungen*. Vortrag auf dem WBO/Busforum Workshop in Karlsruhe, 25.11.2005.

[Schn93] B. Schneier. *Applied Cryptography: Protocols, Algorithms, and Source Code in C*. John Wiley & Sons, New York. 1993.

[Seit02] Ludwig Seitz. *Physikalische Angriffe auf Blockchiffren am Beispiel der NESSIE Kandidaten*. Diplomarbeit, Universität Karlsruhe (TH), 2002.

Index