Versuch V7

C405 Hardwarepraktikum II

Abnahme: 20. Januar 2025
Stand: 7. Januar 2025

Tom Mohr
Martin Ohmeyer

Inhaltsverzeichnis

1 Kodierung **1**
1.1 Notwendigkeit 1
1.2 Besondere Sequenzen 1
1.3 Datenblöcke 2
1.4 Kontrollblöcke 3

2 Implementation **4**
2.1 Parallelisierung 4
2.2 Der Sender als Zustandsautomat 5

3 Verworfene Ansätze **6**

1 Kodierung

1.1 Notwendigkeit

Nacheinander gesendete Nibble müssen auf der Leitung voneinander unterscheidbar sein. Um dies zu gewährleisten, wird ein Taktsignal in den Datenstrom kodiert, indem verhindert wird, dass ein gesendetes Nibble gleich seinem Vorgänger ist. Dies verursacht den Sonderfall zweier gleicher aufeinanderfolgender Nibble. Zu dessen Behandlung muss eine Escape-Sequenz definiert werden, welche die gleichen Nibble durchbricht. Durch die Einführung einer solchen Escape-Sequenz wird ihr eigenes Auftreten im Datenstrom jedoch selbst zu einem Sonderfall. Zur Handhabung dieser Sonderfälle und zur Kennzeichnung von Blöcken wird ein Protokoll zwischen Sender und Empfänger vereinbart.

1.2 Besondere Sequenzen

Aus den in Abschnitt 1.1 dargelegten Gründen werden vordefinierte Sequenzen in den Datenstrom injiziert. Eine solche Sequenz setzt sich aus vier festen (Escape-Sequenz) und vier dynamischen Bits (Handlungsanweisung: "Command") zusammen. Alle verwendeten 4 Bit Sequenzen sind in Tabelle 1.1 gelistet und in ihrer Funktion erläutert.

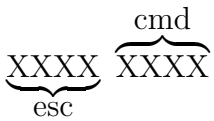


Abb. 1.1: Aufbau einer Kodierungssequenz

Die **Escape-Sequenz** trennt kodierende Sequenzen vom restlichen Datenstrom ab. Sie selbst hält keine Information darüber, um welche Kodierung es sich handelt. Aufgrund ihrer Sonderfunktion darf sie nicht regulär im Datenstrom auftreten und muss ggf. selbst escaped werden.

Commands erhalten erst dann ihre Bedeutung, wenn sie unmittelbar nach der Escape-Sequenz stehen. Sie geben Auskunft darüber, um welche Kodierung es sich handelt und implizieren, wie sich ein Dekodierender verhalten muss, um die originalen Daten wieder zu rekonstruieren. Ist das nachfolgende Nibble auf einen Command (im binären) mit diesem identisch, wird anstelle des normalen Commands dessen Fallback-Version genutzt.

Hex	Bezeichnung	Bedeutung
0	escapeSequence	Das nächste Nibble ist ein Command
1	beginDataBlockDefault	Ein Datenblock beginnt
2	beginDataBlockFallback	Ein Datenblock beginnt
3	beginControlBlockDefault	Ein Kontrollblock beginnt
4	beginControlBlockFallback	Ein Kontrollblock beginnt
5	endBlockDefault	Der aktuelle Block endet
a	endBlockFallback	Der aktuelle Block endet
6	insertPrevNibbleAgainDefault	Ein doppeltes Nibble im Datenstrom
7	insertPrevNibbleAgainFallback	Ein doppeltes Nibble im Datenstrom
8	insertEscSeqAsDataDefault	Die Esc-Seq trat im Datenstrom auf
9	insertEscSeqAsDataFallback	Die Esc-Seq trat im Datenstrom auf

Tabelle 1.1: Bitsequenzen und ihre Bedeutung

1.3 Datenblöcke

Datenblöcke dienen zur Übertragung der Rohdaten. Da gemäß Aufgabenstellung bis zu 1GB große Dateien zu senden sind und maximal 64 Byte pro Paket übertragen werden, ist die Paket-ID $\lceil \log_2 (1\text{GB}/64\text{B}) \rceil = 24$ Bit lang. Der Aufbau eines Datenpakets ist in Abbildung 1.2 dargestellt. Die Länge des gesamten Paketes ist wegen der Kodierung zur Laufzeit möglicherweise größer, als dort abgebildet.

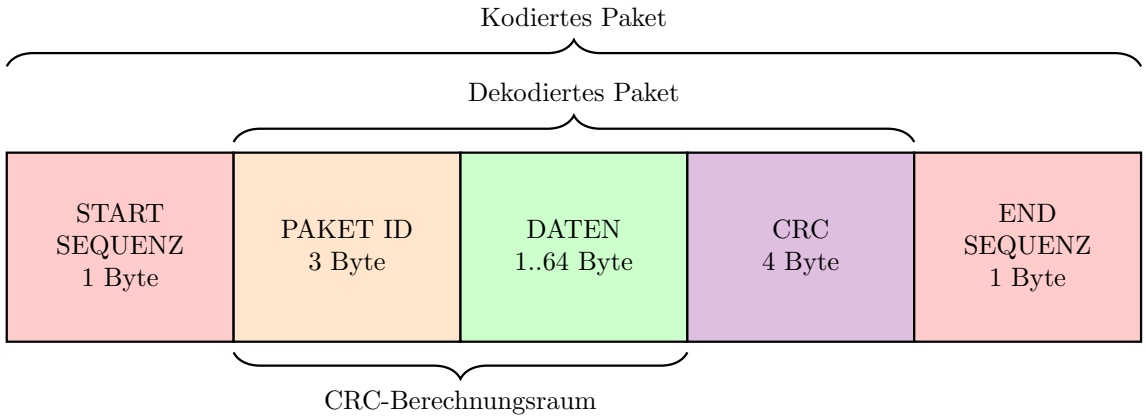


Abb. 1.2: Aufbau eines Datenpakets

1.4 Kontrollblöcke

Kontrollblöcke werden in verschiedenen Kontexten gesendet und übermitteln dem Kommunikationspartner Informationen über den Zustand der Übertragung. So werden sie z.B. genutzt um mitzuteilen, dass man bereit ist zu senden, oder dass man ein Datenpaket korrekt (oder inkorrekt) erhalten hat.

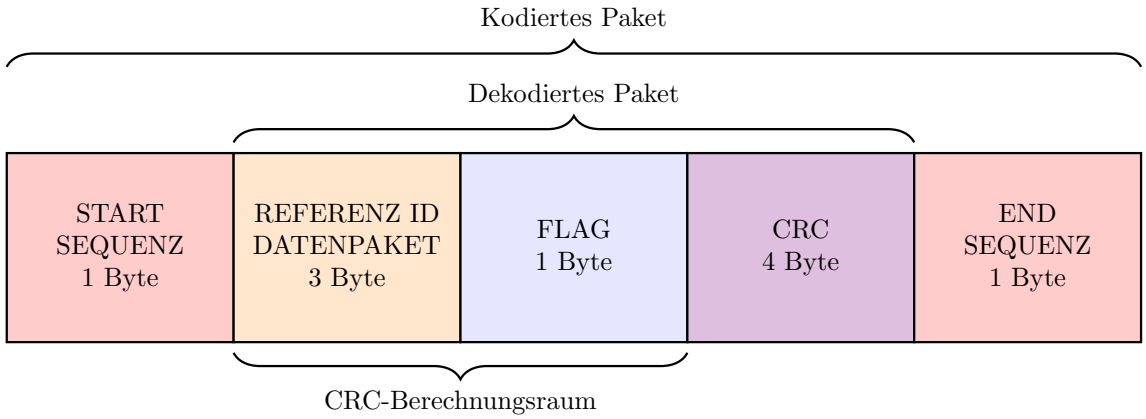


Abb. 1.3: Aufbau eines Kontrollpakets

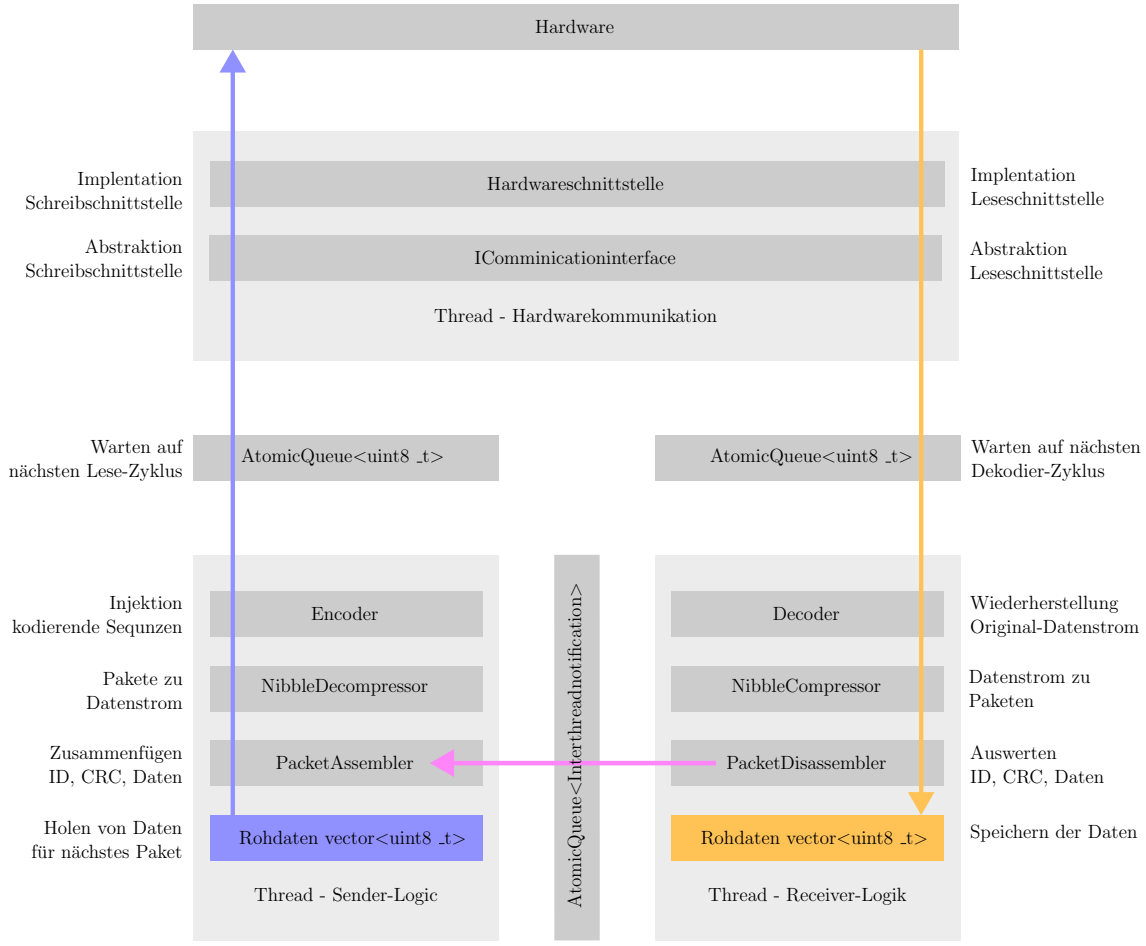
Ein Kontrollpaket setzt sich entsprechend Abbildung 1.3 zusammen. Die referenzierte Datenpaket-ID ist bei allen Kontrollblöcken, welche keine Antworten auf Datenpakete sind, Null. Das Flag Übermittelt die eigentliche Information. Die Länge des gesamten Paketes ist wegen der Kodierung zur Laufzeit möglicherweise größer, als in Abbildung 1.3 dargestellt. Eine Auflistung aller genutzten Flags erfolgt in Tabelle 1.2.

Hex	Bezeichnung	Bedeutung
2	close connection	Alle Daten versandt, breit Verbindung zu schließen
4	resend	Inkorrektes Datenpaket erhalten, neusenden nötig
6	connect	online, bereit Verbindung aufzubauen
8	received	Korrektes Datenpaket erhalten, sende nächstes Paket

Tabelle 1.2: Kontrollflags und ihre Bedeutung

2 Implementation

2.1 Parallelisierung



2.2 Der Sender als Zustandsautomat

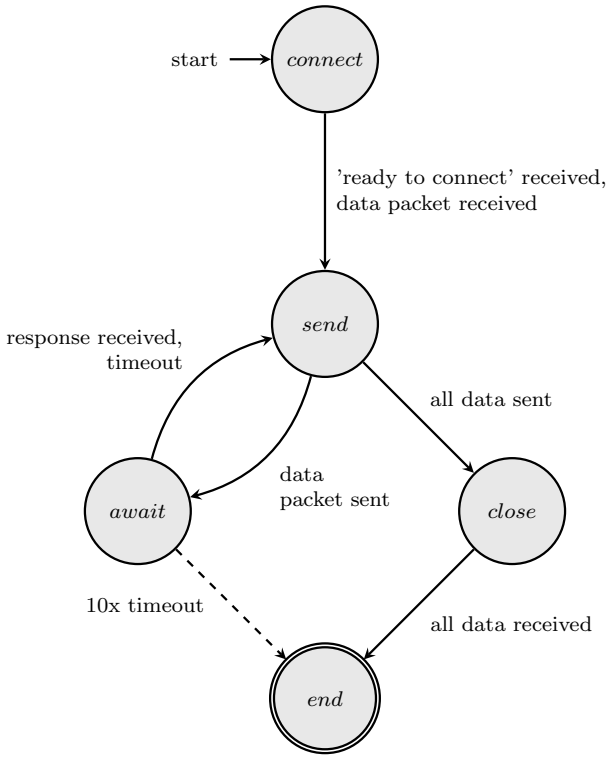


Abb. 2.1: Zustandsautomat des Senders

3 Verworfene Ansätze