6. Sequentielle Grundschaltungen

6.1 Schieberegister

6.1.1 Schieberegisterstrukturen

Schieberegister bestehen aus hintereinandergeschalteten Flipflops. Die Belegung des Dateneingangs wird in das erste Flipflop übernommen, mit dem nächsten Takt in das nächste Flipflop usw. Typische Anwendungen betreffen den Informationstransport, die Signalverzögerung und die Wandlung zwischen serieller Informationsübertragung und paralleler Informationsdarstellung (Serialisierung/Deserialisierung).

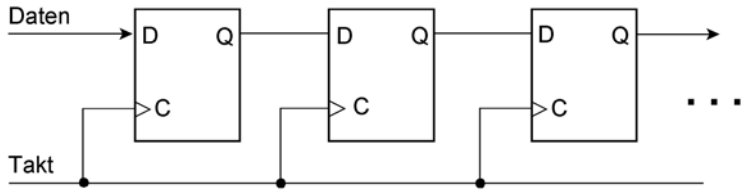

Abb. 6.1 Schieberegister (Grundschaltung). Serielle Eingabe, serielle Ausgabe.

Die Schieberegister unterscheiden sich grundsätzlich in Hinsicht auf

- die Zugänglichkeit der Flipflops (nur serielle Ein- und Ausgabe, parallele Ausgabe, parallele Eingabe),
- die Anzahl der Schieberichtungen (eine oder zwei),
- die Art der Steuerung (Taktsteuerung, Erlaubnissteuerung).

Parallele Ausgabe
Die Daten werden von den Ausgängen der Schiebeflipflops abgenommen.

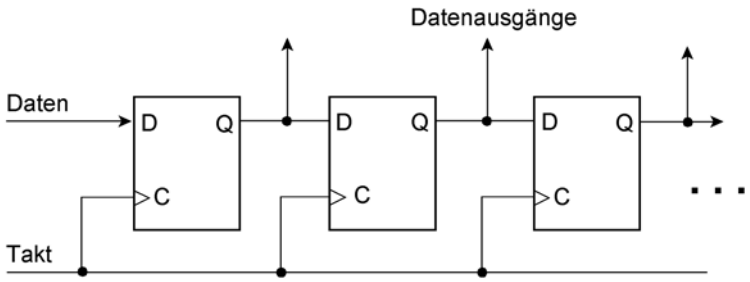

Abb. 6.2 Schieberegister mit paralleler Ausgabe.

Asynchrone parallele Eingabe
Hierzu werden die Setz- und Rücksetzeingänge der Schiebeflipflops ausgenutzt (Abb. 6.68); das asynchrone Laden ist ein durch ein Übernahmesignal gesteuertes Setzen oder Rücksetzen.

Beim Laden verhält sich das Flipflop wie ein Latch..

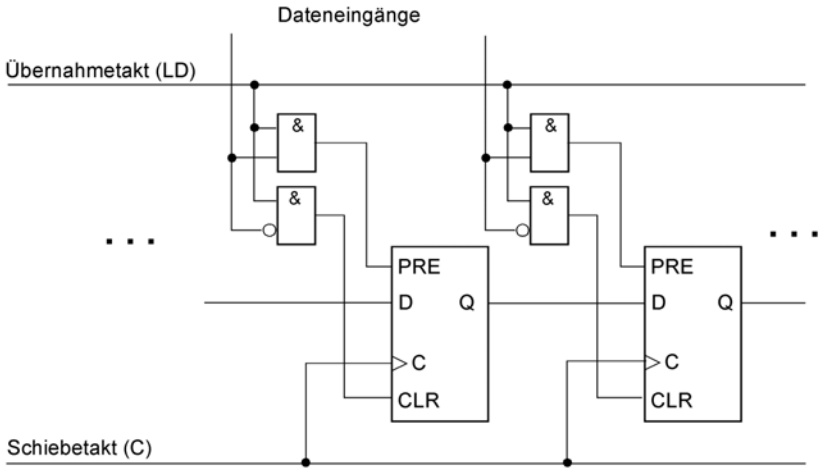

Abb. 6.3 Asynchrone parallele Eingabe.

Synchrone parallele Eingabe

Alle Vorgänge beziehen sich auf denselben Takt. Den Schiebeflipflops sind Auswahlschaltungen vorgeordnet. Zum Schieben wird der Schiebeweg ausgewählt, zum Laden die parallelen Dateneingänge.

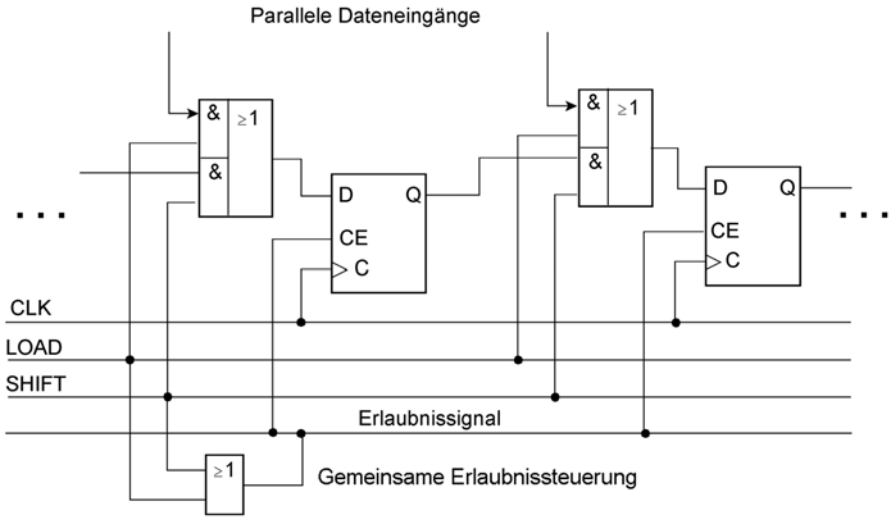

Abb. 6.4 Synchrone parallele Eingabe am Beispiel eines Schieberegisters mit Erlaubnissteuerung. Beim Laden (LOAD) oder Schieben (SHIFT) wird das Erlaubnissignal aktiv[1)].

1: Die Abbildung zeigt lediglich eine Einfachschaltung zur Veranschaulichung des vollsynchronen Ladens und Schiebens. Es gibt keine Vorrangregelung. Sind LOAD und SHIFT gleichzeitig aktiv, ergibt sich eine praktisch unbrauchbare Belegung (Dateneingang ODER vorgeordnetes Flipflop).

Schieberichtungen

Schieberichtungen werden üblicherweise mit bildhaften Begriffen wie links/rechts oder vorwärts/rückwärts bezeichnet. Das klingt einleuchtend, drückt aber nicht immer zutreffend aus, was wirklich abläuft.

Praxistipp: Bei Nutzung fertiger Schieberegister immer genau nachsehen, von welcher Bitposition in welche geschoben wird, bei eigenen Entwürfen die Entwurfsabsicht wirklich exakt formulieren und Bit für Bit durchgehen[2)].

Links und rechts

Diese Richtungsbezeichnungen beziehen sich genaugenommen darauf, dass die im Register gespeicherten Bits als Binärzahlen interpretiert werden[3)], wobei die höherwertigen Stellen links von den jeweils niederwertigen angeordnet sind. Das Linksschieben führt von den niederwertigen zu den höherwertigen, das Rechtsschieben von den höherwertigen zu den niederwertigen Bitpositionen.

Nur eine Schieberichtung

Die Register sind universell einsetzbar. Die Richtungsfrage stellt sich nur dann, wenn auf die Flipflops auch parallel zugegriffen wird. Eine bestimmte Entwurfsabsicht (z. B. Rechtsschieben) wird verwirklicht, indem die die Bitpositionen in der Schaltung entsprechend angeschlossen werden.

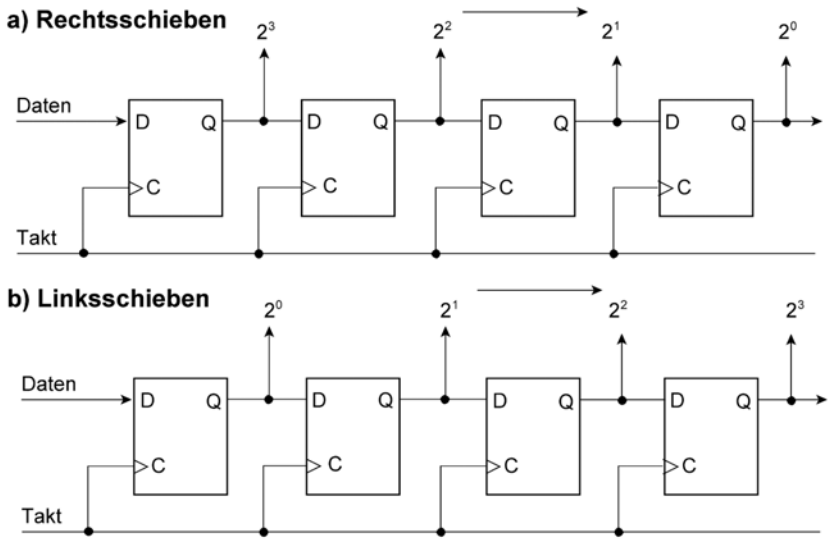

Abb. 6.5 Die Schieberichtung ergibt sich durch entsprechendes Anschließen der Flipflops.

2: Die falsche Schiebereihenfolge ist ein typischer Entwurfs- oder Programmierfehler (beispielsweise beim Programmieren von Zugriffsroutinen für entsprechende Schnittstellen).

3: Jedenfalls sollte es so sein. Nicht alle Autoren von Datenblättern halten sich aber daran. So beziehen sich “links” und “rechts” manchmal nur auf die Anordnung der Flipflops im Schaltbild.

Zwei Schieberichtungen

Das typische Rechts-Links-Schieberegister hat einen Schiebetakt und Steuereingänge für die Richtungssteuerung. Das Umsteuern der Schieberichtung erfordert eine Auswahlschaltung vor den Dateneingängen der Flipflops.

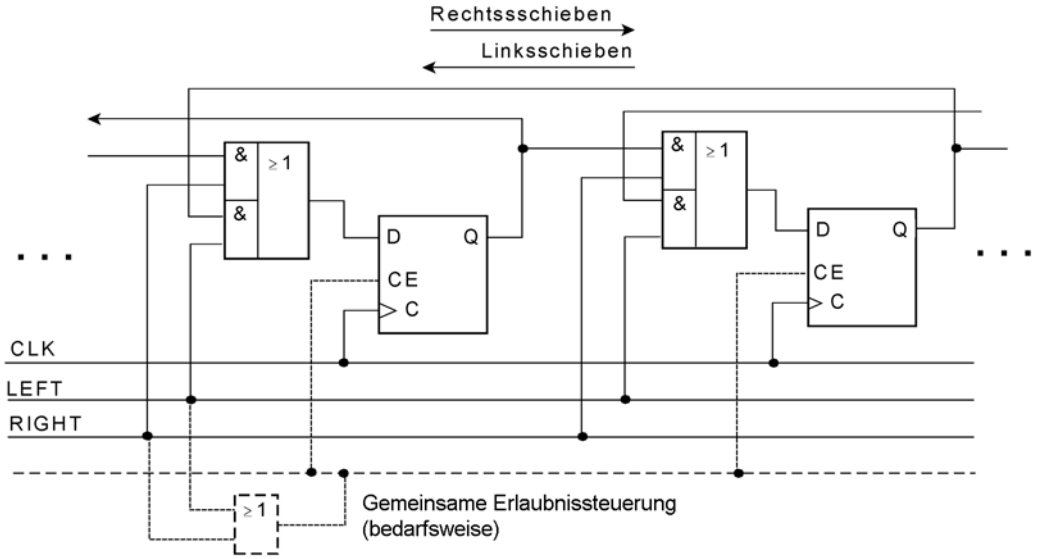

Abb. 6.6 Elementares Schieberegister für beide Schieberichtungen.

Erlaubnissteuerung

Abb. 6.7 zeigt ein einfaches Schieberegister mit Erlaubnissteuerung auf Grundlage von DE-Flipflops. Ist das Erlaubnissignal (SHIFT) inaktiv, bleiben die Daten in den Flipflops erhalten.

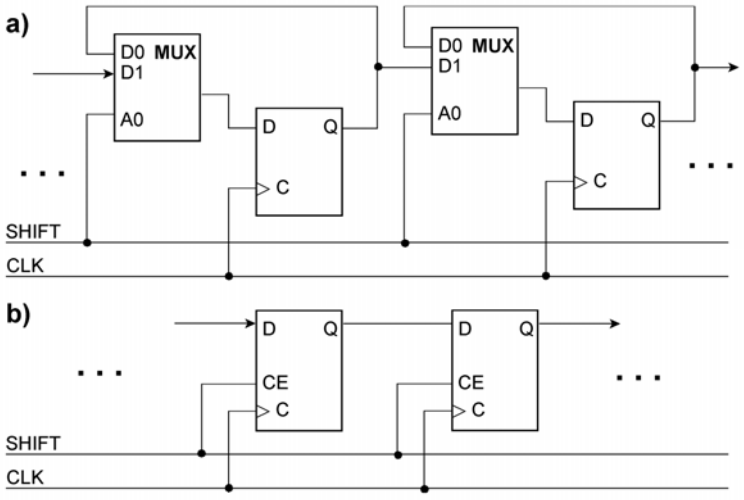

Abb. 6.7 Schieberegister mit Erlaubnissteuerung. a) Grundschaltung mit D-Flipflops; b) mit DE-Flipflops.

Pufferregister

Pufferregister sind erforderlich, wenn die Parallelzugriffe vom Schiebevorgang unabhängig sein sollen. Die Pufferregister können mit transparenten Latches oder mit flankengesteuerten Flipflops aufgebaut sein. Puffer- und Schieberegister können folgendermaßen kombiniert werden:

a) Vorgeschaltetes Pufferregister zur Paralleleingabe. Während die eine Datenbelegung ausgeschoben wird, kann bereits die nächste in das Pufferregister geladen werden. Die typische Anwendung: die sog. Parallel-Serien-Wandlung (Serialisierung; z. B. um Daten Bit für Bit über ein serielles Interface zu transportieren).

b) Nachgeschaltetes Pufferregister zur Parallelausgabe. Es kann gleichsam ein Schnappschuss der Schieberegisterbelegung übernommen werden. Während das Schieben weiterläuft, kann man die übernommenen Daten abtransportieren. Die typische Anwendung: die sog. Serien-Parallel-Wandlung (Deserialisierung; z. B. um Daten zu empfangen, die Bit für Bit über ein serielles Interface geliefert werden).

c) Schieberegister mit ein- und ausgangsseitigen Pufferregistern. Diese Anordnung kann serielle Daten sowohl senden als auch empfangen (Serialisierung und Deserialisierung).

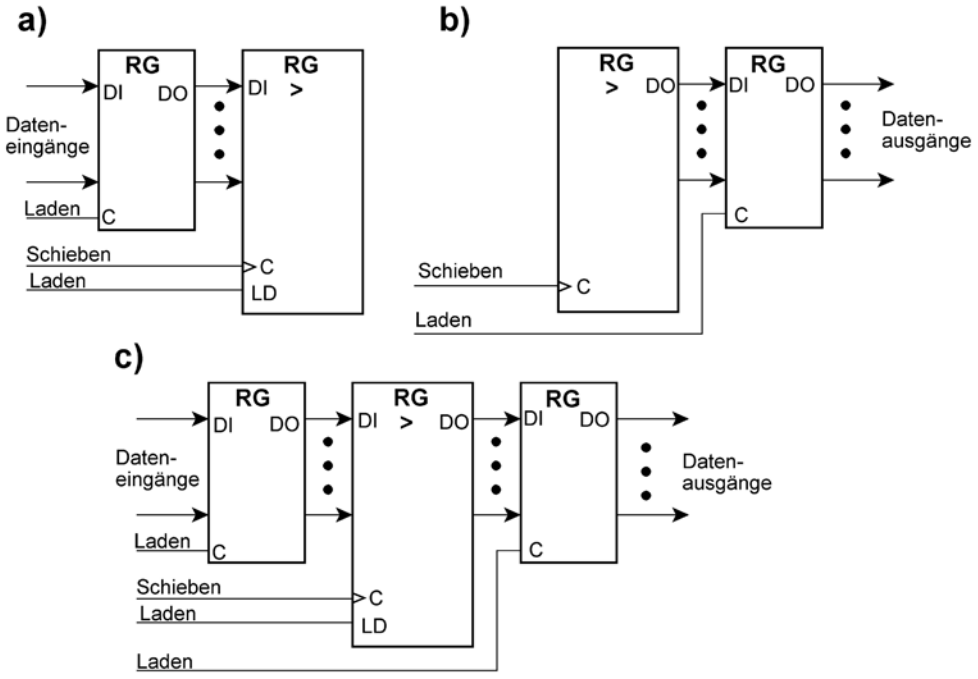

Abb. 6.8 Schieberegister mit Pufferregistern.

6.1.2 Schieberegisterentwurf

Schieberegisterschaltkreise

Die Schieberegisterschaltkreise der herkömmlichen Logikbaureihen sind für Neuentwicklungen nicht zu empfehlen. Aus üblichen universellen Registerschaltkreisen kann man Schieberegister bauen, indem man die Schiebewege auf auf der Leiterplatte verschaltet.

Die parallele Ausgabe ist trivial. Das Problem der parallelen Eingabe kann durch Ausnutzung der typischen Tri-State-Ausgänge gelöst werden. Im Vergleich zu „echten“ Schieberegistern braucht man zwar mehr Schaltkreise, es sind aber Typen aus der Massenfertigung, die es in nahezu allen Logikbaureihen gibt – auch in den ganz modernen. Somit lassen sich Probleme der Pegelwandlung, der Teilabschaltung (Partial Power Down) usw. lösen, indem man jeweils geeignete Typen einsetzt. Die grundsätzliche Alternative zur Eingabe über Schieberegister ist die Abfrage über Multiplexer.

Funktionselemente

Die Entwicklungssysteme für programmierbare und anwendungsspezifische Schaltkreise stellen entsprechende Funktionselemente bereit. Diese Schieberegister können mit durchlaufendem Takt betrieben werden. CE ist das Erlaubnissignal für die Schiebefunktion, L das Ladesteuersignal (es wirkt unabhängig von CE). Das Laden hat Vorrang vor dem Schieben.

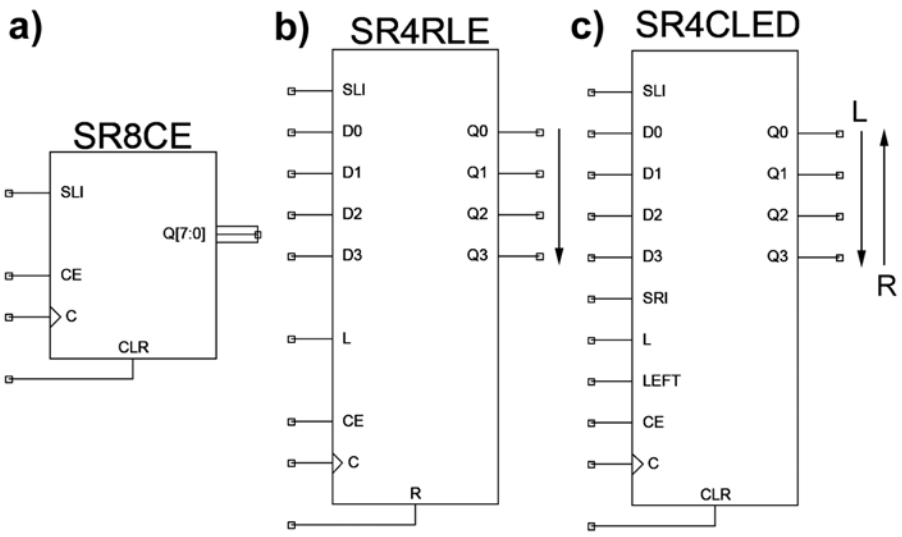

Abb. 6.9 Schieberegister als Funktionselemente. Eine kleine Auswahl (nach [S04]). a) acht Bits mit paralleler Ausgabe; b) vier Bits mit paralleler Ein- und Ausgabe; c) vier Bits mit paralleler Ein- und Ausgabe und zwei Schieberichtungen (bidirektional). Die Pfeile geben die Schieberichtungen an.

Funktionselemente kaskadieren

Schieberegister mit paralleler Ausgabe werden kaskadiert, indem man den – in Schieberichtung gesehen – letzten Datenausgang mit dem jeweils nachfolgenden Schiebedateneingang verbindet.

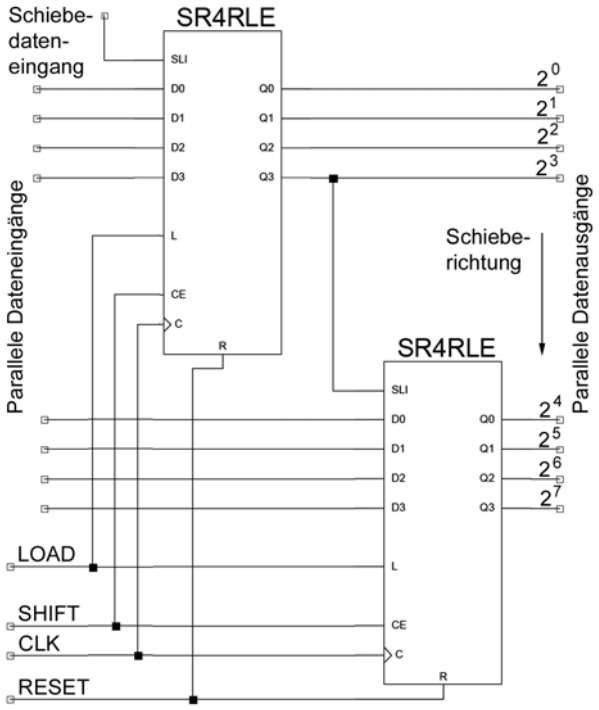

Abb. 6.10 Schieberegister kaskadieren (1). Eine Schieberichtung. Mit den angegebenen Stellenwerten handelt es sich um ein Linksschieben.

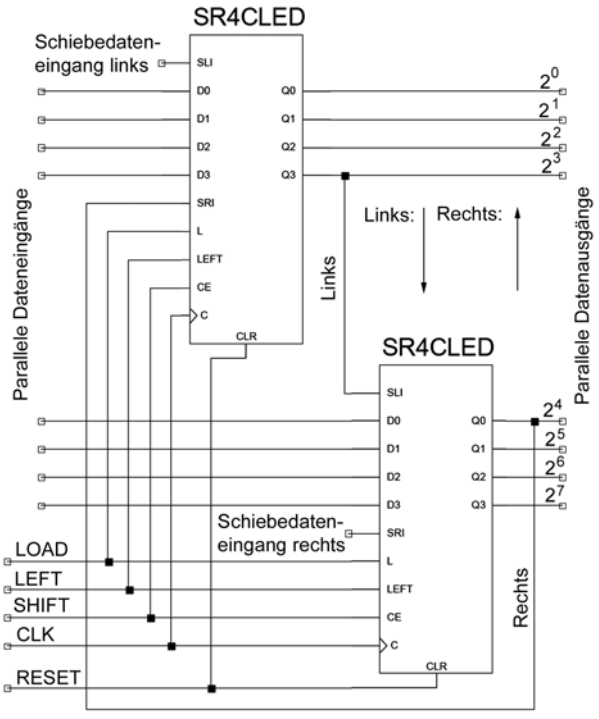

Abb. 6.11 Schieberegister kaskadieren (2). Zwei Schieberichtungen. Das Linksschieben geht in Richtung der höherwertigen Bitpositionen (von 0 nach 1 usw.), das Rechtsschieben in Richtung der niederwertigen (von 3 nach 2 usw).

Von Grund auf entwerfen

Das Schieberegister fügt sich am besten in moderne Entwurfslösungen ein, wenn es mit einem durchlaufenden Takt betrieben wird (vollsynchrone Auslegung mit Erlaubnissteuerung). Es kann systematisch auf der RTL-Ebene oder als Mehrfunktionsregister entworfen werden. Auf der RTL-Ebene werden Speicher- und Schiebefunktionen voneinander getrennt. Das Schiebenetzwerk wird als Funktionszuordner entworfen. Auf diese Weise können u. a. auch Verschiebungen um mehr als ein Bit erledigt werden. Betrachtet man das Schieberegister als vollsynchrones Mehrfunktionsregister, so bereiten die typischen Schiebe- und Ladefunktionen nur wenig Mühe. Auch ist es nicht schwierig, das Schieben mit weiteren Funktionen zu kombinieren (Beispiel: ein Zähler mit bitseriellem Zugang zum Laden und Abfragen).

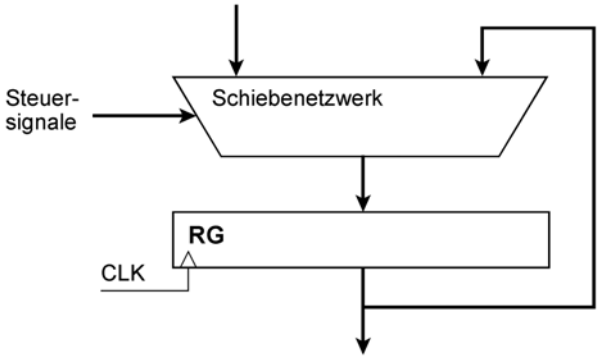

Abb. 6.12 Schieberegisterentwurf auf der RTL-Ebene.

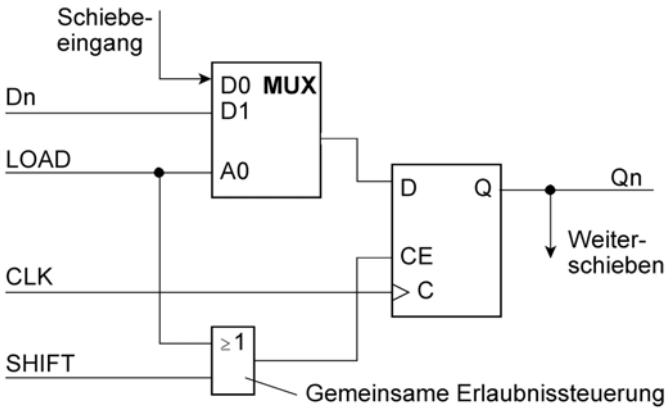

Abb. 6.13 Beispiel 1: parallele Ein- und Ausgabe, eine Schieberichtung (vgl. Abb. 6.75b). Das Laden hat Vorrang vor dem Schieben.

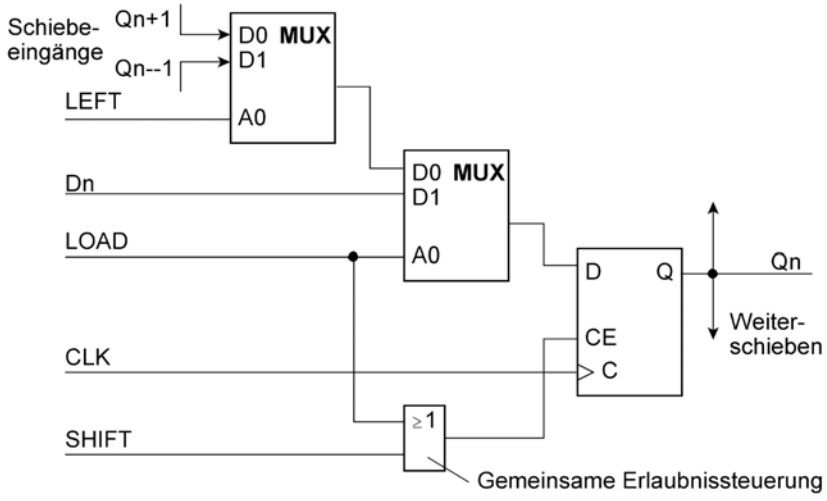

Abb. 6.14 Beispiel 2: parallele Ein- und Ausgabe, zwei Schieberichtungen (vgl. Abb. 6.75c). Das Laden hat Vorrang vor dem Schieben. SHIFT ist die allgemeine Schiebeerlaubnis. Rechtsschieben bei LEFT = 0, Linksschieben bei LEFT = 1.

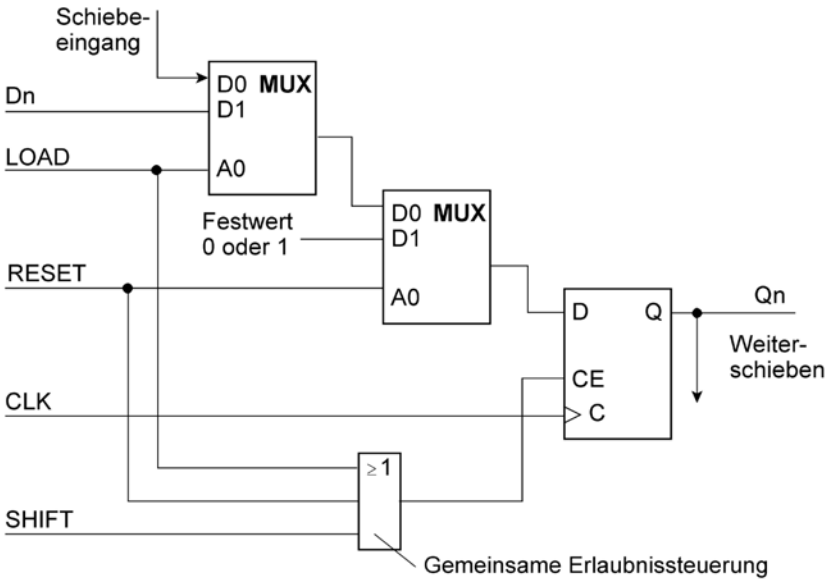

Abb. 6.15 Beispiel 3: Wie Beispiel 1, aber mit synchronem Rücksetzen, wobei die Flipflops mit einem beliebigen Bitmuster geladen werden können. Es ist als Festwert an den zweiten Multiplexer (Pfeil) anzulegen.

6.2 Zähler und Teiler

6.2.1 Zähler und Teiler als Zustandsautomaten

Zähler und Teiler sind Zustandsautomaten, die ihre Ausgangssignale in Abhängigkeit von der Anzahl der Taktimpulse bilden. Sie nehmen mit jedem Zähltakt einen anderen Zustand an und weisen dabei eine starre Folge von Zustandsübergängen auf. Der aktuelle Zustand entspricht der Anzahl der wirksam gewordenen Taktimpulse (Zählerstand). Zähler (Counter) und Teiler (Divider) unterscheiden sich lediglich darin, ob der Zählerstand auf Ausgänge geführt ist oder nicht. Deshalb werden im Folgenden Zähler und Teiler weitgehend gemeinsam behandelt. Dabei wird "Zähler" als Allgemeinbegriff verwendet.

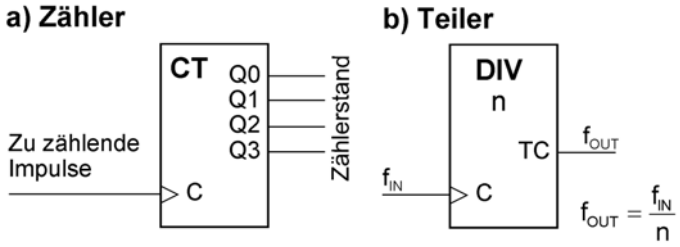

Abb. 6.16 Zähler und Teiler. a) Zähler. Der Zählerstand ist auf die Ausgänge geführt. b) Teiler. Der Ausgang signalisiert lediglich, dass alle Zustände durchlaufen wurden und ein Übergang zum Anfangszustand stattfindet. TC = Endzustand (Terminal Count). Die Impulsfolgefrequenz am Ausgang (f_{OUT}) entspricht der eingangsseitigen Taktfrequenz (f_{IN}) geteilt durch das Teilerverhältnis n (Frequenzteilung).

Codierung

Mit n Flipflops lassen sich maximal 2^n verschiedene Zustände codieren und somit auch abzählen. Die Codierung des Zählers bestimmt, welche Flipflopbelegungen bei aufeinander folgenden Taktimpulsen gebildet werden.

Abb. 6.17 Zähler und Teiler.

Zählweite, Zählen modulo n

Das sind Bezeichnungen dafür, dass sich nach jeweils n Taktimpulsen der Zählablauf wiederholt. Hat ein Zähler mit n Stellungen (also n Zählzustände) so sind n Taktimpulse erforderlich, um wieder in den Anfangszustand zu gelangen. Beispiel: der Zählablauf 0 – 1 – 2 – 3 – 4 – 0 – 1 usw. Es erfordert offensichtlich fünf Taktimpulse, um beispielsweise – mit Zustand Null beginnend – den Zustand Null erneut zu erreichen (Zählweite Fünf; modulo-5-Zähler).

Teilungsfaktor, Teilungsverhältnis

Beide Angaben kennzeichnen die Frequenzteilung durch zyklisches Abzählen von Taktimpulsen.

Teilungsfaktor n

Alle n Eingangsimpulse wird ein Ausgangsimpuls abgegeben.

Teilungsverhältnis 1:n

Eine Impulsfolgefrequenz f_{IN} am Eingang führt zu einer Impulsfolgefrequenz $f_{OUT} = \frac{f_{IN}}{n}$ am Ausgang (Frequenzteilung 1: n).

Zähler sind Zustandsautomaten

Herkömmlicherweise wird oft nur der Binär- oder Dezimalzähler als Zähler im eigentlichen Sinne angesprochen. Diese Auffassung ist aber fachlich nicht exakt und erschwert das Verständnis moderner Schaltungslösungen. Wichtig sind zunächst allein die Anzahl der Zustände und die Zustandsübergänge, unabhängig von der Zustandscodierung. Ein modulo-n-Zähler ist ein Zustandsautomat mit n Zuständen und folgenden Merkmalen:

- Jeder Takt veranlasst einen Zustandswechsel zu einem eindeutig bestimmten Folgezustand.
- Nach n Takten wird wieder der Anfangszustand eingenommen.
- Dieser einfache Ablauf (fortlaufendes zyklisches Zählen) kann folgendermaßen abgewandelt werden:
  - Anhalten in einem beliebigen Zustand (Wartebedingung),
  - Rückführung in den Ausgangszustand (Löschen, Rücksetzen),
  - Umkehrung der Zählrichtung (vorwärts/rückwärts oder aufwärts/abwärts),
  - Einstellen eines beliebigen Zustandes (Laden).

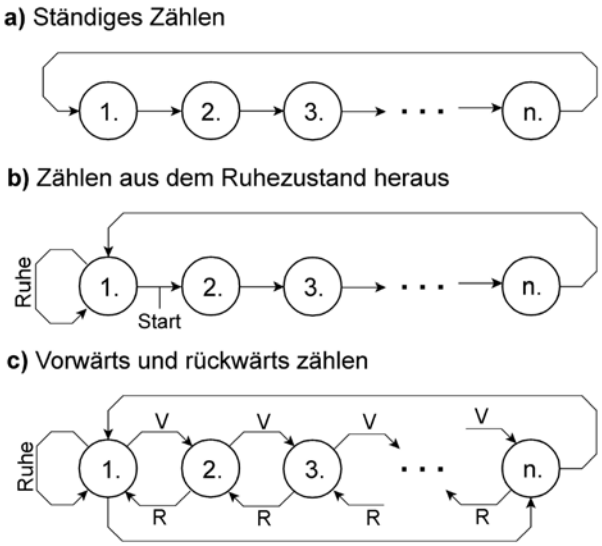

Abb. 6.18 Der modulo-n-Zähler als Zustandsautomat (anhand von drei Beispielen).

6.2.2 Zählerentwurf

Elementare Zählschaltungen

Solche Schaltungen (Ringzähler, Johnsonzähler, Binärzähler usw.) werden in den folgenden Abschnitten näher erläutert. Tabelle 6.1 gibt einen ersten Überblick über wichtige Eigenschaften.

Grundschaltung	Maximale Zählweite mit n Flipflops	Maximale Zählfrequenz
Ringzähler	n	Maximale Taktfrequenz der Flipflops
Johnsonzähler	2 n	Maximale Taktfrequenz der Flipflops
Asynchroner Binärzähler	2^n	Maximale Taktfrequenz der Flipflops. Aber Schaltverzögerung der Binärstellen gemäß O(n)
Synchroner Binärzähler	2^n	Abhängig von der Verzögerungszeit der Zählnetzwerke und damit von der Anzahl der Flipflops

Tabelle 6.1 Typische Zählschaltungen.

Ring- und Johnsonzähler kommen nur für sehr geringe Zählweiten in Betracht (Richtwert: höchstens etwa 20 bis 50). Für größere Zählweiten ist eine binäre Codierung der Zählzustände unumgänglich. Beispiel: Zählen modulo 2^{20} = 1 048 576. Ein Ringzähler würde 1 048 576 Flipflops erfordern, ein Johnsonzähler 524 288, ein Binärzähler 20.

Zählerschaltkreise

Die Zählerschaltkreise der herkömmlichen Logikbaureihen sind für Neuentwicklungen nicht zu empfehlen. Deshalb werden die Einzelheiten dieser Zählerschaltkreise auch nicht näher behandelt.

Funktionselemente

Die Entwicklungssysteme für programmierbare und anwendungsspezifische Schaltkreise stellen entsprechende Funktionselemente bereit. Diese Zähler können mit durchlaufendem Takt betrieben werden. CE ist das Erlaubnissignal für die Zählfunktion, L das Ladesteuersignal (es wirkt unabhängig von CE). Das Laden hat Vorrang vor dem Zählen. Es gibt Typen mit synchronem Rücksetzen (R) und und asynchronem Löschen (CLR). Die Rücksetzfunktion hat Vorrang vor dem Laden. Die TC-Signale werden aktiv, wenn der Zählerstand dem höchsten Zählwert entspricht (Endzustand). Die CEO-Signale dienen zum Nachschalten (Kaskadieren) weiterer Zähler.

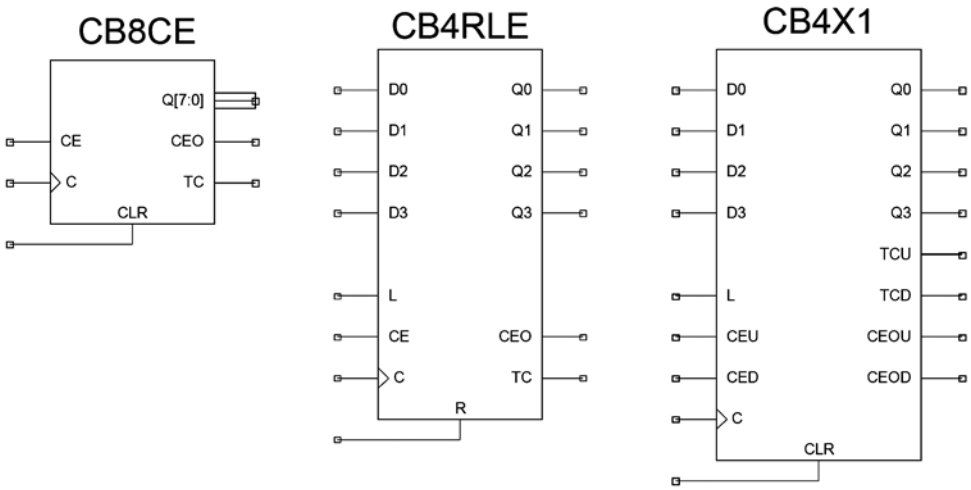

Abb. 6.19 Zähler als Funktionselemente. Eine kleine Auswahl. a) Binärzähler mit acht Bits; b) ladbarer Binärzähler mit vier Bits; c) ladbarer Vorwärts-Rückwärts-Binärzähler mit vier Bits.

Zähler kaskadieren

Kaskadieren heißt, Zähler so hintereinanderzuschalten, dass der nachgeordnete Zähler immer dann eine Zustandswechsel ausführt, wenn der vorgeordnete alle Zählzustände durchlaufen hat und wieder in den Anfangszustand einläuft. Typische Anwendungsfälle:

- Aus Funktionselementen mit geringerer Zählweite ist ein Zähler mit größerer Zählweite aufzubauen (beispielsweise ein 24-Bit-Zähler aus sechs 4-Bit-Zählern).
- Komplexere Aufgaben der Ablaufsteuerung. Beispielsweise soll nach jedem Durchlauf eines ersten Steuerzählers ein zweiter Steuerzähler um einen Schritt weiterschalten.
- Mehrstellige Zähler auf Grundlage eines Stellenwertsystems (z. B. Dezimalzähler). Eine Zählweite n erfordert in einem Stellenwertsystem zur Basis B $\log_B n$ Zähler modulo B.

Beispiel: Ein Dezimalzähler, der modulo 1000 zählen soll, wird aus $\log_{10} 1000 = 3$ modulo-10-Zählern aufgebaut.

Um Zähler kaskadieren zu können, müssen sie Ausgangssignale bereitstellen, die den jeweiligen Endzustand anzeigen. Ist ein solches Endesignal aktiv, so veranlasst der nächste Zähltakt, dass der betreffende Zähler seinen Anfangszustand einnimmt. Gleichzeitig muss der nachgeschaltete Zähler in den jeweiligen Folgezustand übergehen, mit anderen Worten, um Eins weiterzählen. Beim Kaskadieren handelt es sich im Grunde darum – in Entsprechung zum Addieren und Subtrahieren – einen Übertrag in die jeweils nachfolgende Zählerstufe einzuspeisen. Deshalb werden die entsprechenden Signale auch als Übertragssignale bezeichnet (Carry beim Vorwärtszählen, Borrow beim Rückwärtszählen). Die Grundlagen der Kaskadierung gelten für alle Zählerarten gleichermaßen, die Schaltungseinzelheiten hängen vom Typ des Zählers ab. Die im Folgenden beschriebenen Schnittstellen beruhen auf Zählerlaubniseingängen (CE), Zählerlaubnisausgängen (CEO) und Taktausgängen (CO).

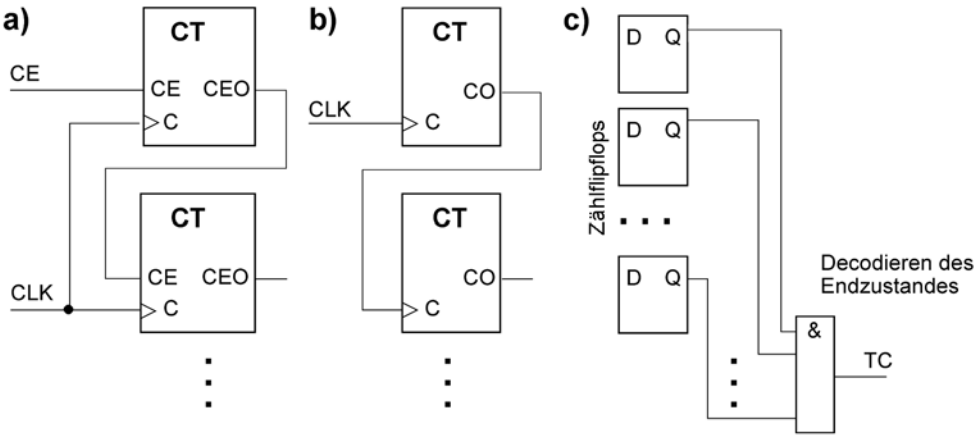

Abb. 6.20 Grundschaltungen der Kaskadierung von Zählern. a) synchrone, b) asynchrone Kaskadierung. c) Decodieren des Endzustandes (Terminal Count TC).

Synchrone Kaskadierung

Alle Zähler haben einen gemeinsamen Takt. Der zweite Zähler darf nur dann zählen, wenn der erste seinen Endzustand erreicht hat. Ist ein dritter Zähler nachgeschaltet, so darf dieser nur dann zählen, wenn beide vorgeordneten Zähler ihren Endzustand erreicht haben, ein vierter Zähler darf nur dann zählen, wenn die drei vorgeordneten Zähler ihren Endzustand erreicht haben usw. Da an allen Zählern der durchlaufende Takt anliegt, dürfen sie überhaupt nur zählen, wenn die allgemeine Zählerlaubnis aktiv ist (Erlaubnissignal CE). Es gilt also

$$CE2 = CE \bullet TC1$$
$$CE3 = CE \bullet TC1 \bullet TC2$$
$$CE4 = CE \bullet TC1 \bullet TC2 \bullet TC3 \text{ usw.}$$

Um das Zusammenschalten der Funktionselemente zu vereinfachen, kaskadiert man oftmals auch diese UND-Verknüpfungen und baut sie in die Zähler ein. Solche Funktionselemente haben Zählerlaubnisausgänge (Count Enable Output CEO). Ihre Signalbelegung entspricht der UND-Verknüpfung des Zählerlaubniseingangs und des Endesignals:

$$CEO = CE \bullet TC$$

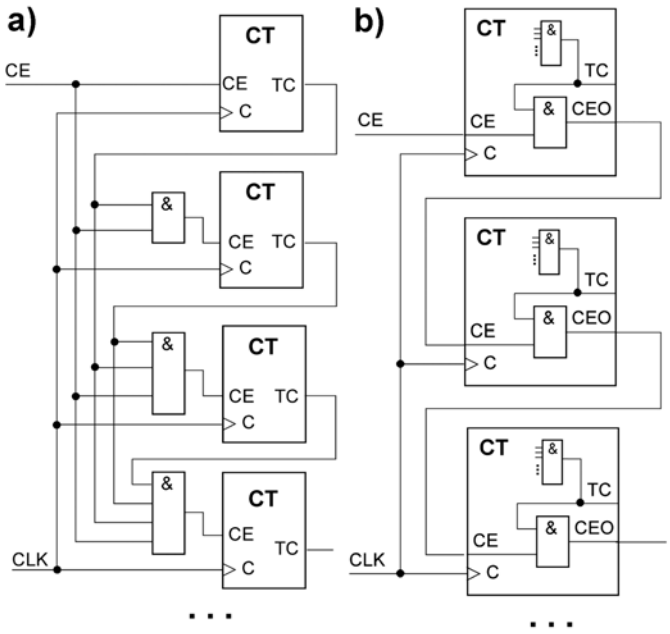

Abb. 6.21 Synchrone Kaskadierung. Die Endzustandssignale aller jeweils vorgeschalteten Zähler müssen konjunktiv verknüpft werden. Da der Takt immer läuft, ist auch die allgemeine Zählerlaubnis (CE) in die Verknüpfung einzubeziehen. b) die übliche Auslegung mit eingebauten kaskadierten UND-Verknüpfungen.

Asynchrone Kaskadierung

Der vorgeordnete Zähler gibt beim Übergang vom End- in den Anfangszustand ein Taktsignal ab, das den nachfolgenden Zähler weiterschaltet (Clock Out / Carry Out CO). Wichtig ist, dass dieses Signal keine Glitches enthält (denn sonst zählt der nachgeschaltete Zähler falsch). Zumeist muss auch die schaltende Flanke des ausgangsseitigen Taktes mit der des eingangsseitigen zeitlich zusammenfallen, damit sich die gesamte Anordnung – von Gatterverzögerungszeiten abgesehen – wie ein einziger langer Zähler verhält.

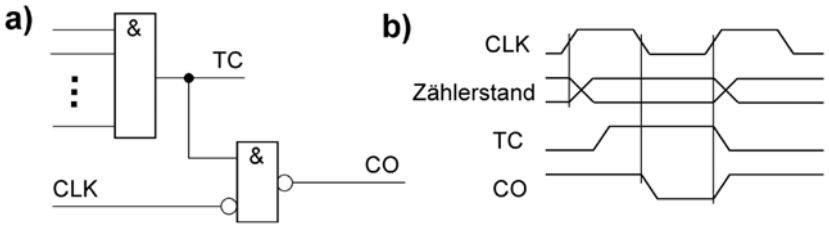

Abb. 6.22 Bildung des Taktausgangssignals (CO) der asynchronen Kaskadierung. Im Beispiel wirkt der Takt mit der Low-High-Flanke. Kehrt der Pegel von CLK auf Low zurück, hat sich schon der neuen Zählerstand eingestellt und TC führt eine gültige Belegung*. Da alle Schaltvorgänge abgeklungen sind, kann die NAND-Verknüpfung keine Glitches ergeben.

Zähler initialisieren und laden

Beim Initalisieren und Laden ist der Zähler im Grunde nichts Anderes als ein gewöhnliches Register. Beide Funktionen können synchron oder asynchron implementiert werden. Die übliche Form der Initialisierung ist das Erzwingen des Anfangszustandes* (Rücksetzen, Löschen).

Der Zähler als Zustandsautomat

Zählende Zustandsautomaten können durch vereinfachte Zustandsfolgetabellen beschrieben werden. Da beim Zählen jeder Zustand genau einen Nachfolger hat, genügt eine Liste der aufeinander folgenden Bitmuster. Der Zähler besteht aus Flipflops mit einem vorgeschalteten Funktionszuordner. Auf dieser Grundlage kann man Zählschaltungen entwerfen, die beliebige Folgen von Bitmustern ausgeben können. Ein typisches Anwendungsbeispiel sind Steuerzähler (Sequencer) für Ablaufsteuerungen.

	C	B	A	TC	TB	TA	DC	DB	DA	SC	SB	SA	RC	RD	RA
0	0	0	0			!			!			!			
1	0	0	1		!			!	!		!				
2	0	1	1		R	!		!							!
3	0	1	0	!			!	!		!					
4	1	1	0			!	!	!	!			!			
5	1	1	1		!		!		!					!	
6	1	0	1		R	!	!								!
7	1	0	0	!									!		

Tabelle 6.2 Die Zustandsfolgetabelle einer Zählschaltung. Links die eigentliche Zustandfolgetabelle, rechts die Ansteuerung verschiedener Flipfloptypen (T-Flipflops, D-Flipflops und RS-Flipflops). Die Ausrufungszeichen kennzeichnen die Stellungen, in denen der jeweilige Flipflopeingang mit einer Eins anzusteuern ist (jedes Ausrufungszeichen entspricht einem Produktterm). Die Ansteuerung des T-Flipflops in Bitposition B wird nachfolgend genauer erläutert.

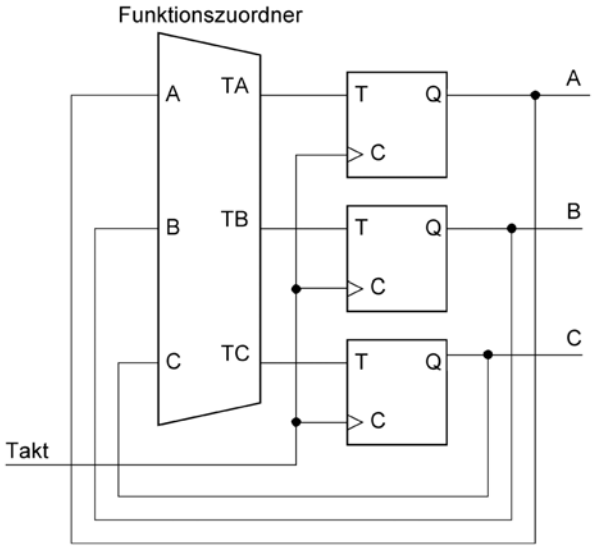

Abb. 6.23 Ein als Zustandsautomat entworfener Zähler. Hier mit T-Flipflops.

Aus der Zustandsfolgetabelle können die Ansteuerbedingungen für die Flipflops – mit anderen Worten, die Booleschen Gleichungen des Funktionszuordners – mühelos abgelesen werden. Sie hängen vom Flipfloptyp ab:

- Ein T-Flipflop ist dann zu schalten, wenn sie die Belegung der betreffenden Bitposition mit dem nächsten Takt ändert.
- Ein D-Flipflop ist dann mit einer Eins zu belegen, wenn im nächsten Takt eine Eins am Ausgang erscheinen soll.
- Ein RS-Flipflop ist zu setzen, wenn mit dem nächsten Takt eine Eins erscheinen soll. Es ist zurückzusetzen, wenn mit dem nächsten Takt eine Null erscheinen soll.

Praxistipp zur Flipflopauswahl: Betrachtet man die Anzahl der Ausrufungszeichen in Tabelle 6.2, so fällt auf, dass sich bei Nutzung der D-Flipflops die umfangreichsten und bei Nutzung der RS-Flipflops die einfachsten Gleichungen ergeben. Die RS-Funktion muss aber in den üblichen programmierbaren Schaltkreisen durch kombinatorische Zusatzbeschaltung verwirklicht werden. Zudem sind es zwei Gleichungen je Flipflop. T-Flipflops haben beim Zählerentwurf den Vorteil, dass man mit einem Steuereingang auskommt und dass die Ansteuerbedingungen zumeist einfacher sind als bei der Nutzung von D-Flipflops. Deshalb beruhen viele Zählschaltungen auf T-Flipflops. In manchen programmierbaren Schaltkreisen hat man die T-Funktion auf Transistorebene verwirklicht, so dass die Schaltzeiten geringer sind als die von rückgekoppelten D-Flipflops, deren Rückkopplungswege über Makrozellen oder Funktionszuordner geführt sind.

Ansteuergleichungen aufstellen

Als Beispiel soll das T-Flipflop in Bitposition B dienen. Das Flipflop muss beim Übergang von Stellung 1 nach Stellung 2 und von Stellung 5 nach Stellung 6 umschalten. Damit das

Umschalten mit dem jeweils nächsten Taktimpuls erfolgen kann, sind somit die Bitmuster der Stellungen 1 und 5 zu decodieren. Betrachtet man auf diese Weise alle drei Bitpositionen in Tabelle 6.4, so ergeben sich für die Toggle-Flipflops folgende Gleichungen:

$$TA = \overline{C} \cdot \overline{B} \cdot \overline{A} \vee \overline{C} \cdot B \cdot A \vee C \cdot B \cdot \overline{A} \vee C \cdot \overline{B} \cdot A$$
$$TB = \overline{C} \cdot \overline{B} \cdot A \vee C \cdot B \cdot A$$
$$TC = \overline{C} \cdot B \cdot \overline{A} \vee C \cdot \overline{B} \cdot \overline{A}$$

Sind die Booleschen Gleichungen gefunden worden, ist das Problem an sich gelöst – den Rest erledigt das Entwicklungssystem.

Rückwärts zählen
Soll rückwärts gezählt werden, so ist die Tabelle von unten nach oben abzuarbeiten. Im Falle des T-Flipflops in Bitposition B ist beim Übergang von Stellung 6 nach Stellung 5 und von Stellung 2 nach Stellung 1 umzuschalten. Die zu decodierenden Stellungen sind in Tabelle 6.2 mit einem “R” gekennzeichnet.

Initialisierung
Falls erforderlich, ist der Anfangszustand einzustellen (Rücksetzwirkung), oder es ist durch Hinzufügen von Bedingungen zu veranlassen, dass der Zähler aus jeder beliebigen Anfangsbelegung in einen gültigen Zustand einlaufen kann.

Der zählende Zustandsautomat als Bitmustergenerator
Eine Zählschaltung mit n Flipflops kann alle 2^n möglichen Bitmuster in beliebiger Reihenfolge durchlaufen. Es darf sich aber kein Bitmuster wiederholen. Stellungen, die nie durchlaufen werden, stellen Don`t-Care-Bedingungen dar, die ggf. zur Schaltungsminimierung ausgenutzt werden können.

Der Zähler als Mehrfunktionsregister
Diese Auslegung liegt dann nahe, wenn neben dem Zählen weitere Funktionen auszuführen sind (Laden, Wartezustände usw.). Die Abb. 6.24 und 6.25 veranschaulichen zwei Beispiele. Es ist jeweils eine Bitposition dargestellt. Die Zählnetzwerke ergeben sich aus den nachfolgenden Abschnitten.

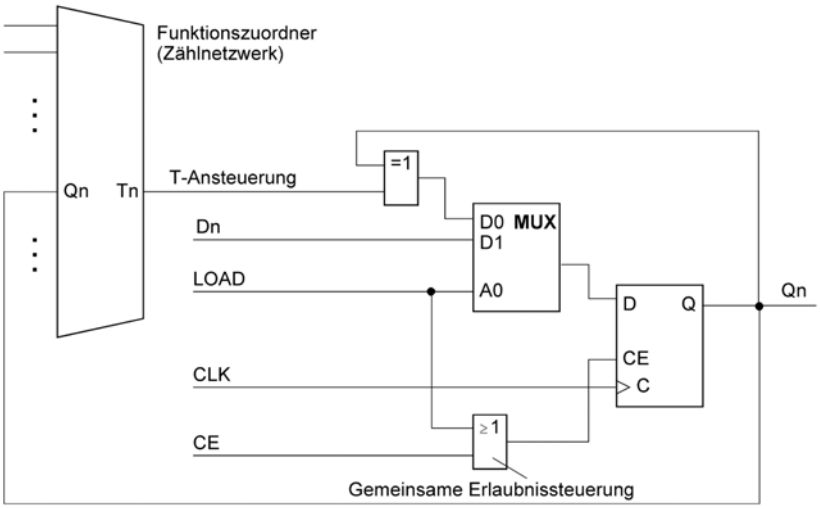

Abb. 6.24 Zähler als Mehrfunktionsregister (1). Die Funktionen: Laden (LOAD), Zählen (CE) und Daten halten. Das DE-Flipflop wird beim Zählen über die Rückführung und die Antivalenzverknüpfung als T-Flipflop betrieben. Ein bitserieller Zugriffweg zum Laden und zum Abholen des aktuellen Zählerstandes ergibt sich, wenn man die Dateneingänge (Dn) mit den Flipflopausgängen (Qn) im Sinne eines Schieberegisters zusammenschaltet.

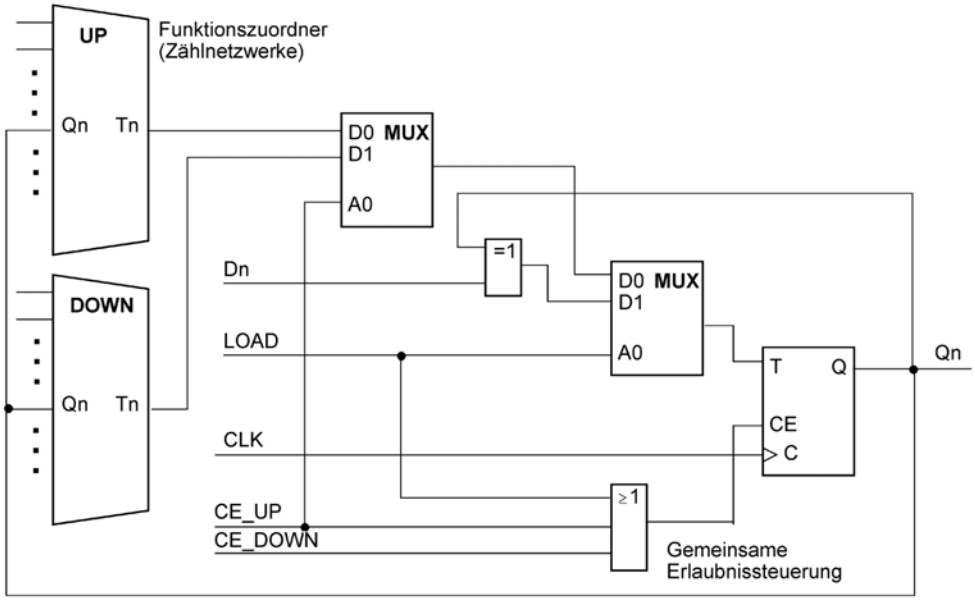

Abb. 6.25 Zähler als Mehrfunktionsregister (2). Die Funktionen: Laden (LOAD), Vorwärtszählen (CE_UP), Rückwärtszählen (CE_DOWN) und Daten halten. Hier wird ein TE-Flipflop eingesetzt, das beim Laden über die Rückführung und die Antivalenzverknüpfung als D-Flipflop betrieben wird.

Der Zähler als Rechenschaltung

Auf der RTL-Ebene werden Speicher- und Zählfunktionen voneinander getrennt (Abb. 6.103). Das Vorwärtszählen ist ein Addieren, das Rückwärtszählen ein Subtrahieren einer

Eins. Das Zählnetzwerk wird so zum vereinfachten Addierer. Auf diese Weise können u. a. auch Zählschritte größer Eins eingeführt werden. Weitere Ergänzungen können beispielsweise das Vergleichen mit vorgegebene Werten und das Laden betreffen. Die Grenzen zum universellen Rechenwerk (Arithmetik-Logik-Einheit ALU) sind fließend.

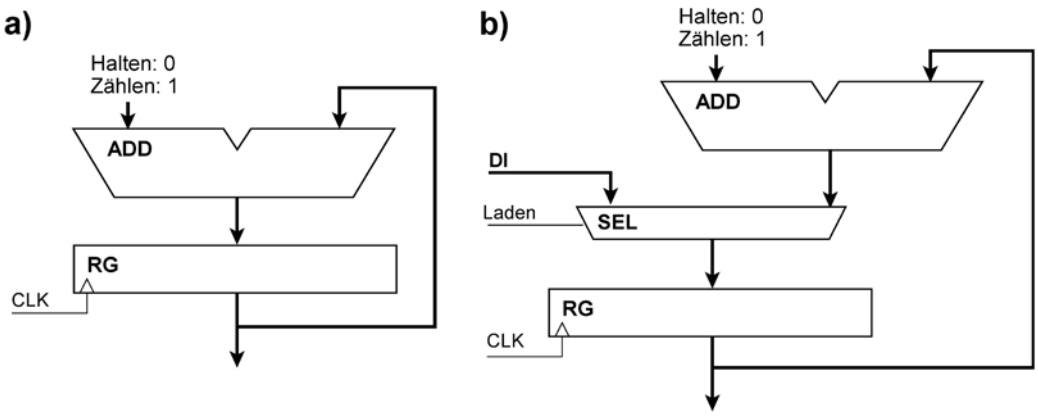

Abb. 6.26 Der Zähler als Rechenschaltung. Zählerentwurf auf der RTL-Ebene. a) Zählen durch Addieren einer Eins. Soll der Zählwert bei laufendem Takt gehalten werden, wird eine Null addiert. b) Erweiterung durch Ladefunktion.

6.2.3 Ringzähler

Ringzähler zählen mit n Flipflops im 1-aus-n-Code modulo n. In jedem Zustand ist ein Flipflop aktiv, während die anderen inaktiv sind. Ein solcher Zähler ist im Grunde ein rückgekoppeltes Schieberegister. Das umlaufende Bitmuster muss anfänglich eingestellt werden (Einschaltrücksetzen). Wird eines der Flipflops gesetzt, und werden alle anderen zurückgesetzt, so läuft eine Eins um.

Selbsteinschwingende Ringzähler

Wird die Rückführung mittels eine NAND- oder NOR-Verknüpfung der ersten n – 1 Flipflops gebildet, so schwingt sich die Schaltung nach spätestens n Taktimpulsen selbst ein. Die NOR-Rückführung ergibt eine umlaufende Eins (1-aus-n-Codierung), die NAND-Rückführung eine umlaufende Null (invertierte 1-aus-n-Codierung).

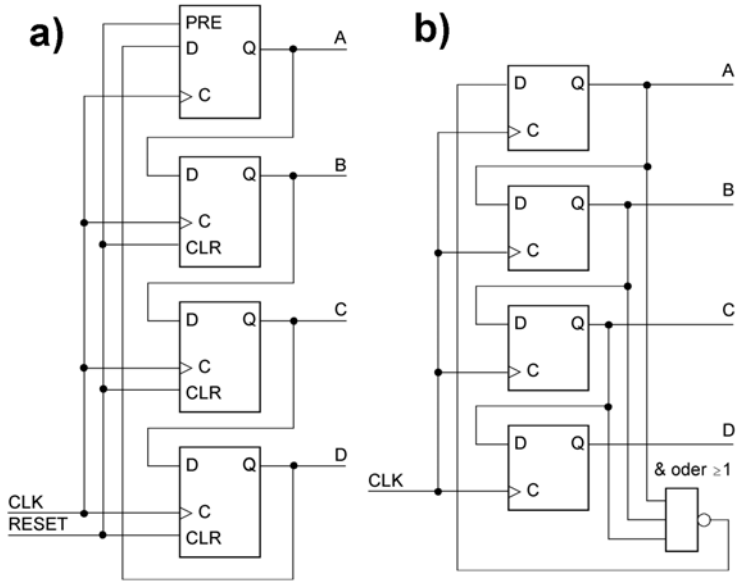

Abb. 6.27 Ringzähler. a) Grundschaltung; b) selbsteinschwingend. NOR-Verknüpfung für positive, NAND-Verknüpfung für negative Ausgangsimpulse.

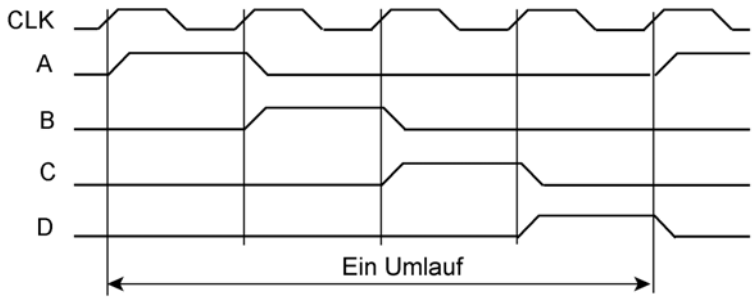

Abb. 6.28 Impulsdiagramm eines Ringzählers.

	A	**B**	**C**	**D**
0	1	0	0	0
1	0	1	0	0
2	0	0	1	0
3	0	0	0	1

Tabelle 6.3 Zustandsfolgetabelle eines Ringzählers.

Die Ausgangssignale weisen keine Glitches auf. Sie können als Takt- oder Strobe-Impulse verwendet werden. Der Ringzähler wird deshalb oft zur Taktaufbereitung und Ablaufsteuerung eingesetzt.

Ringzähler steuern
Ringzähler lassen sich steuern, indem man DE-Flipflops einsetzt und deren CE-Eingänge miteinander verbindet (Zählerlaubnis).

6.2.4 Johnsonzähler

Johnsonzähler zählen mit $^{n}/_{2}$ Flipflops modulo n. Dazu wird das Ringzählerprinzip abgewandelt:

- Der Ausgang des letzten Flipflops wird invertiert auf den Eingang des ersten zurückgeführt.
- Alle Flipflops werden anfänglich gelöscht.

Es ergibt sich ein typisches Impulsmuster, wobei die Ausgänge mit jeweils einer Taktperiode Versatz ein- und in der gleichen Reihenfolge wieder ausschalten (Johnson-Code). Da sich mit jedem Takt nur eine Ausgangssignalbelegung ändert, können die Belegungen des Johnsonzählers glitchfrei decodiert werden.

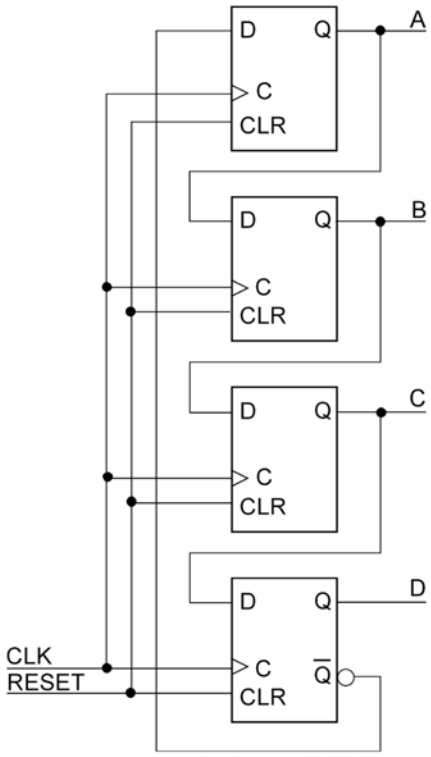

Abb. 6.29 Johnsonzähler. a) Grundschaltung; b) selbsteinschwingend und auf beliebige Zählweiten einstellbar.

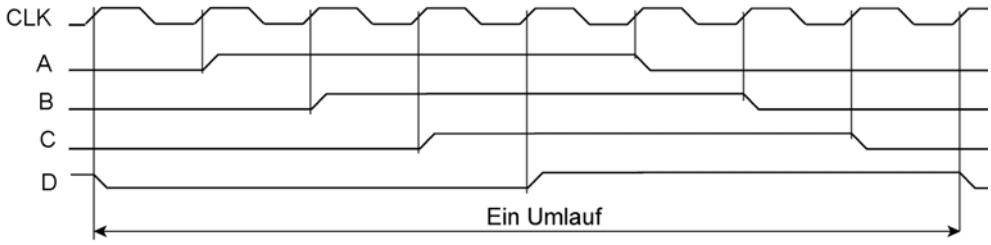

Abb. 6.30 Impulsdiagramm des Johnsonzählers.

	A	B	C	D	Decodierung 1 aus n
0	0	0	0	0	$\overline{A} \cdot \overline{D}$
1	1	0	0	0	$A \cdot \overline{B}$
2	1	1	0	0	$B \cdot \overline{C}$
3	1	1	1	0	$C \cdot \overline{D}$
4	1	1	1	1	$C \cdot A$
5	0	1	1	1	$\overline{A} \cdot B$
6	0	0	1	1	$\overline{B} \cdot C$
7	0	0	0	1	$\overline{C} \cdot D$

Tabelle 6.4 Zustandsfolgetabelle des Johnsonzählers.

Johnsonzähler oder Ringzähler?
Braucht man Ausgangssignale im 1-aus-n-Code, dürfte in den meisten Fällen der Ringzähler zweckmäßiger sein. Er kann für beliebige Zählweiten und zum Selbstanschwingen eingerichtet werden. In allen anderen Einsatzfällen (beispielsweise dann, wenn man nur eine Teilerfunktion braucht oder wenn nur wenige Zählerstellungen zu signalisieren sind) müsste die Auswahl ggf. anhand von Probeentwürfen erfolgen.

6.2.5 Asynchrone Binärzähler

Beim binären Zählen ist der Zählwert eine Binärzahl, zu der in jedem Zählschritt eine Eins addiert oder subtrahiert wird (Addition beim Vorwärts-, Subtraktion beim Rückwärtszählen). Die Zustandsfolgetabelle entspricht den Variablenbelegungen in einer üblichen Wahrheitstabelle. Beim Vorwärtszählen wird sie von oben nach unten, beim Rückwärtszählen von unten nach oben durchlaufen. Nach dem Erreichen des jeweiligen Endwertes führt der nächste Zählschritt wieder auf den Anfangswert.

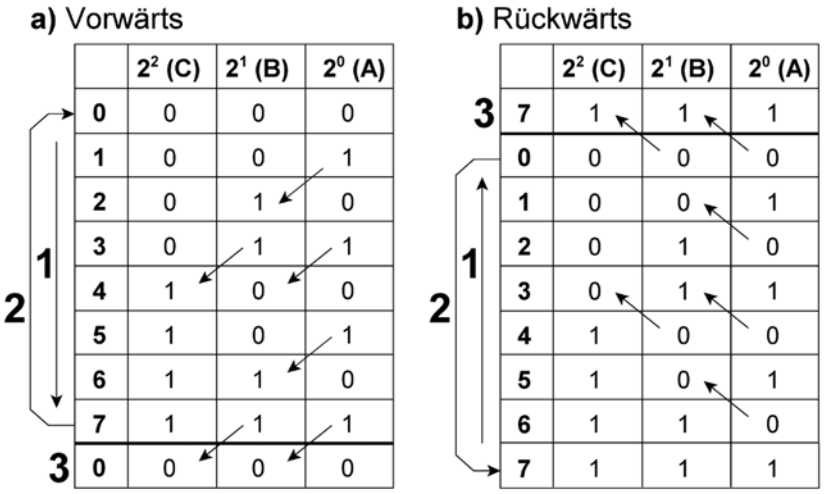

Abb. 6.31 Das Zählschema des Binärzählers (anhand von drei Stellen).1 - Zählrichtung; 2 - das Umschlagen vom End- auf den Anfangswert (Wrap Around); 3 - wiederholte Darstellung des Anfangswertes, um das Zählen beim Umschlagen zu veranschaulichen.

Die Funktionsweise des Asynchronzählers ist aus Abb. 6.112 leicht zu erschließen.

a) Vorwärtszählen:

- Die niedrigstwertige Stelle (Flipflop A) schaltet mit jedem Zählimpuls (0 – 1– 0 – 1 usw.). Das entspricht direkt der Schaltweise eines T-Flipflops.
- Die nächsthöhere Stelle (Flipflop B) schaltet immer dann, wenn Flipflop A von 1 nach 0 schaltet.
- Die wiederum nächsthöhere Stelle (Flipflop C) schaltet immer dann, wenn Flipflop B von 1 nach 0 schaltet usw.

b) Rückwärtszählen:

- Die niedrigstwertige Stelle (Flipflop A) schaltet mit jedem Zählimpuls (0 – 1– 0 – 1 usw.). Das entspricht direkt der Schaltweise eines T-Flipflops.
- Die nächsthöhere Stelle (Flipflop B) schaltet immer dann, wenn Flipflop A von 0 nach 1 schaltet.
- Die wiederum nächsthöhere Stelle (Flipflop C) schaltet immer dann, wenn Flipflop B von 0 nach 1 schaltet usw.

Ein Zähler ergibt sich somit als einfache Hintereinanderschaltung von T-Flipflops. Dabei wird der Takteingang an den Ausgang des vorhergehenden Flipflops angeschlossen. Vorwärtszähler schalten mit der High-Low-Flanke des Taktes, Rückwärtszähler mit der Low-High-Flanke.

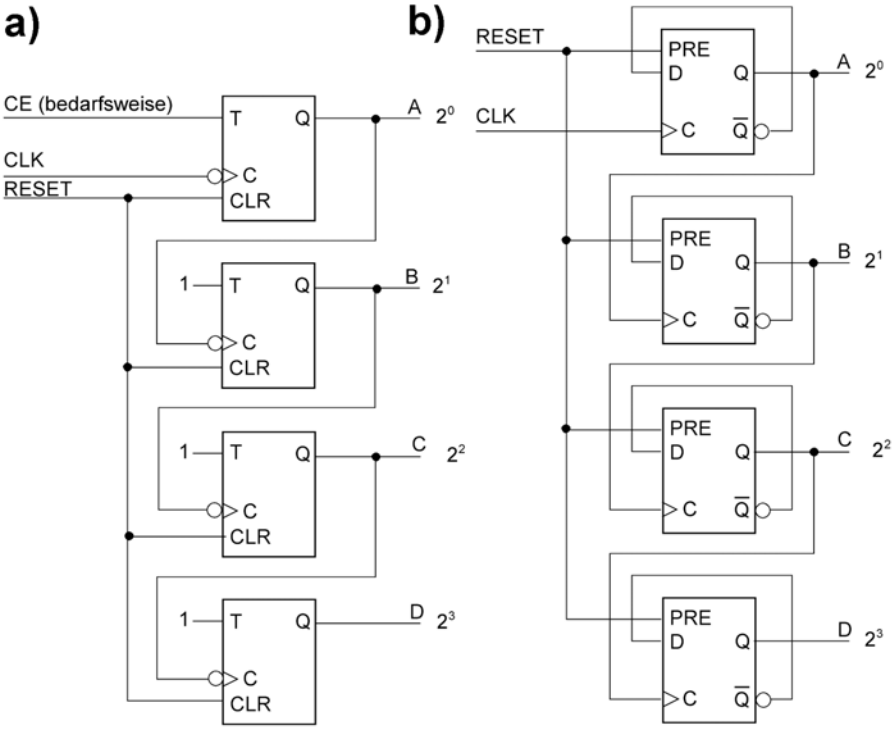

Abb. 6.32 Asynchronzähler (Ripple Counter). Typische Grundschaltungen. a) Vorwärtszähler mit T-Flipflops. Der T-Eingang des ersten Flipflops kann genutzt werden um den Zähler zu steuern. b) Rückwärtszähler mit rückgekoppelten D-Flipflops. Das Rücksetzen ist nur dann erforderlich, wenn vom Anfangszustand an gezählt werden soll.

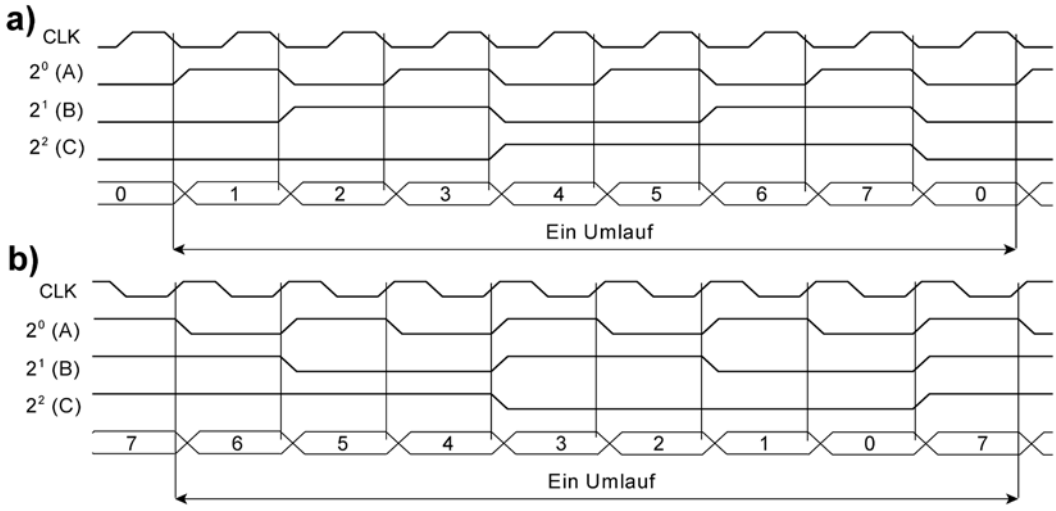

Abb. 6.33 Impulsdiagramme von dreistelligen asynchronen Binärzählern. a) vorwärts, b) rückwärts zählend.

Die Zählfrequenz
Gibt es keine Steuerung über den T-Eingang, ist auf nichts Rücksicht zu nehmen. Das erste Flipflop kann dann mit der maximalen Schaltfrequenz (Toggle Frequency) gemäß Datenblatt betrieben werden. Das nächste Flipflop schaltet dann mit der Hälfte, das übernächste mit einem Viertel der eingangsseitigen Zählfrequenz usw. Diese Tatsache ermöglicht es, längere Zähler in Abschnitte einzuteilen, die mit verschiedenen Technologien implementiert werden. Manchmal kann man das Weiterzählen in den höherwertigen Binärstellen auch programmseitig erledigen (Mikrocontroller).

Der einzelne Zählschritt ist erst dann beendet, wenn auch das letzte Flipflop geschaltet hat. Ein Asynchronzähler aus n Flipflops mit der Verzögerungszeit (Takt zu Daten) t_{pd} braucht somit für einen Zählschritt (Zählperiode) mindestens eine Zeit

$$t_{cp\_min} = n \cdot t_{pd}$$

Die maximale Zählfrequenz eines derart eingesetzten Asynchronzählers nimmt also mit der Anzahl der Stellen ab. Entfällt die Auswertung nach jedem Zählschritt, kommen die Vorteile des Asynchronzählers zur Wirkung – einfachster Aufbau, höchste Zählfrequenz und minimale Stromaufnahme (weil Taktflanken immer nur dann auftreten, wenn es zum Weiterzählen erforderlich ist). Ein typischer Einsatzfall ist die Frequenzteilung.

Asynchronzähler kaskadieren
Der Ausgang des höchstwertigen Flipflops liefert den Takt für den jeweils nachgeschalteten Zähler.

6.2.6 Synchrone Binärzähler

Alle Flipflops sind mit dem Zähltakt beschaltet. Die Ansteuerbedingungen können aus der Zustandsfolgetabelle abgelesen werden:

a) Vorwärtszählen:

- Die Bitposition 2^0 schaltet mit jedem Takt.
- Die Bitposition 2^1 schaltet dann, wenn die Bitposition 2^0 mit Eins belegt ist.
- Die Bitposition 2^2 schaltet dann, wenn die Bitpositionen 2^1 und 2^0 mit Eins belegt sind usw.

Die Ansteuerbedingung eines T-Flipflops in einer beliebigen Bitposition 2^n ($n > 0$) ergibt sich also als konjunktive Verknüpfung der Flipflopausgänge aller niederwertigen Bitpositionen:

$$T_n = Q_{n-1} \cdot Q_{n-2} \cdot \ldots \cdot Q_0$$

b) Rückwärtszählen:

- Die Bitposition 2^0 schaltet mit jedem Takt.
- Die Bitposition 2^1 schaltet dann, wenn die Bitposition 2^0 mit Null belegt ist.
- Die Bitposition 2^2 schaltet dann, wenn die Bitpositionen ^{21} und 2^0 mit Null belegt sind usw.

Die Ansteuerbedingung eines T-Flipflops in einer beliebigen Bitposition 2^n ($n > 0$) ergibt sich als konjunktive Verknüpfung der invertierten Flipflopausgänge oder als NOR-Verknüpfung aller niederwertigen Bitpositionen:

$$T_n = \overline{Q_{n-1}} \cdot \overline{Q_{n-2}} \cdot \ldots \cdot \overline{Q_0} = \overline{Q_{n-1} \vee Q_{n-2} \vee \ldots \vee Q_0}$$

Zusammengefasst: beim Vorwärtszählen schaltet eine Bitposition dann um, wenn alle jeweils niederwertigen Bitpositionen mit Einsen, beim Rückwärtszählen dann, wenn sie mit Nullen belegt sind.

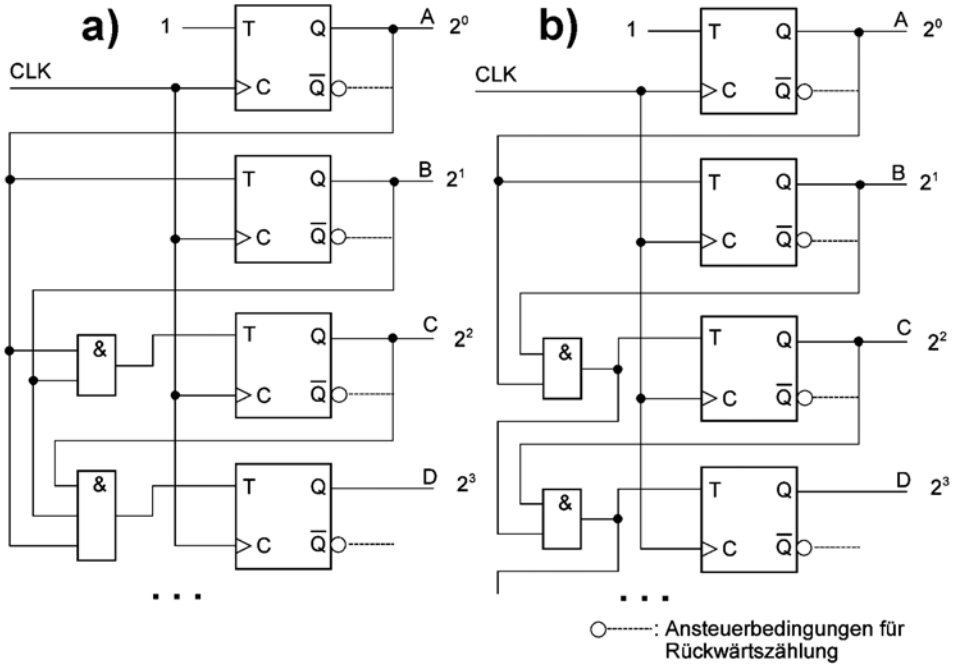

Abb. 6.34 Synchrone Binärzähler. Grundschaltungen ohne Vorkehrungen zum Rücksetzen, Steuern, Kaskadieren usw. a) Ansteuerung mit Schaltungstiefe 1. In jeder nachfolgenden Stufe hat das UND-Gatter einen Eingang mehr. b) UND-Verknüpfungen durch Kaskadierung von UND-Gattern mit zwei Eingängen. Beim Rückwärtszählen werden die Ansteuerbedingungen mit den invertierten Ausgängen der Flipflops gebildet.

Es liegt nahe, die T-Ansteuerbedingungen der Flipflops mit UND-Gattern zu bilden. Die Anzahl der Eingänge nimmt mit jeder Bitposition um Eins zu. Weil die UND-Verknüpfung assoziativ ist, können UND-Gatter mit vielen Eingängen durch Kaskadierung von UND-Gattern mit wenigen Eingängen aufgebaut werden. Im Extremfall kommt man je Bitstelle mit einem UND-Gatter mit zwei Eingängen aus, dessen Ausgang auf einen Eingang des nachfolgenden UNDs geführt wird.

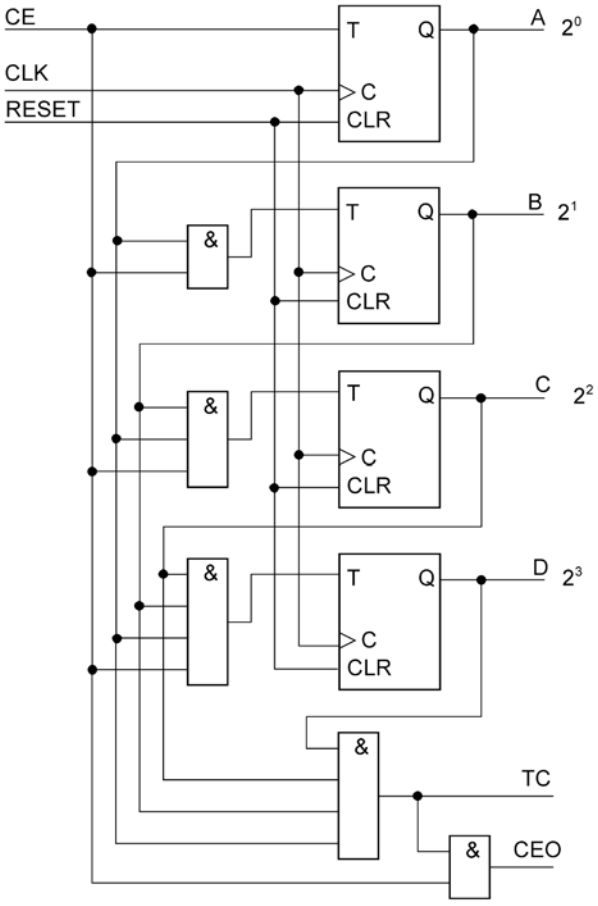

Abb. 6.35 Ein generischer synchroner Binärzähler. Weitere Funktionen können eingebaut werden, indem man den T-Eingängen der Flipflops entsprechende Auswahlschaltungen vorschaltet (Mehrfunktionsregister).

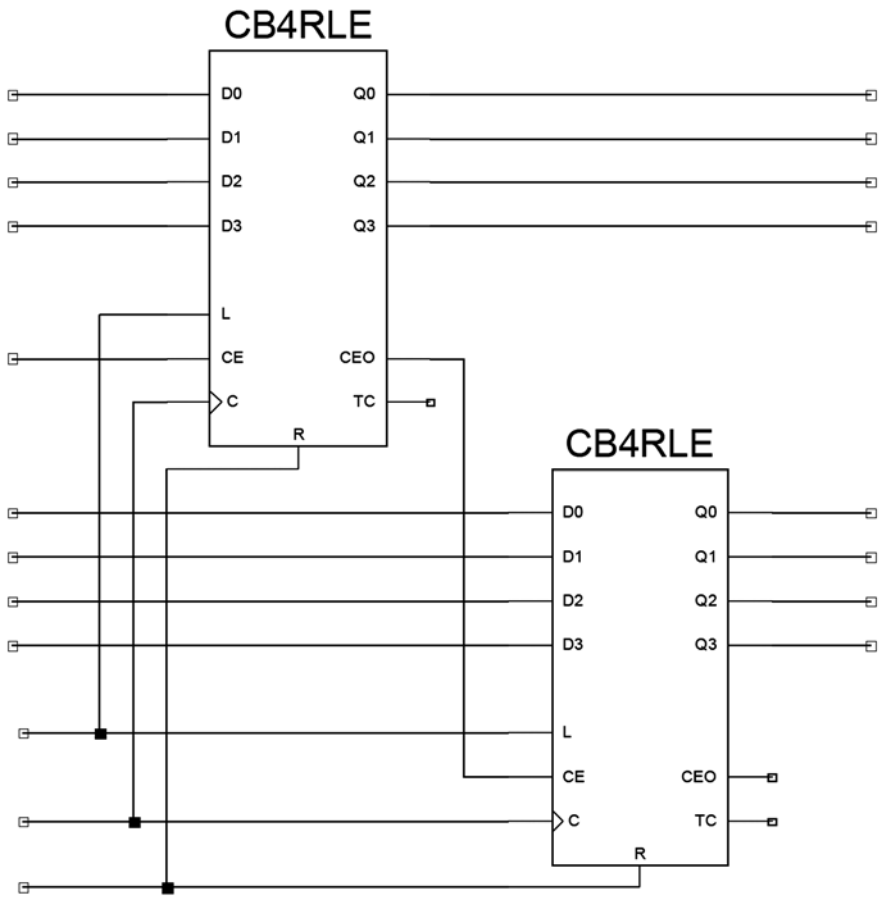

Abb. 6.36 Kaskadierung generischer Binärzähler ([4]). Um den Endzustand (hier: Ausgangsbelegung FFH) zu decodieren, sind die TC-Ausgänge konjunktiv zu verknüpfen.

6.2.7 Vorwärts und rückwärts zählen

Rückwärtszählen heißt, die Folge der Zählzustände in umgekehrter Richtung zu durchlaufen. Umsteuerbare Zähler arbeiten mit einem gemeinsamen Takt oder mit verschiedenen Takten für beide Zählrichtungen.

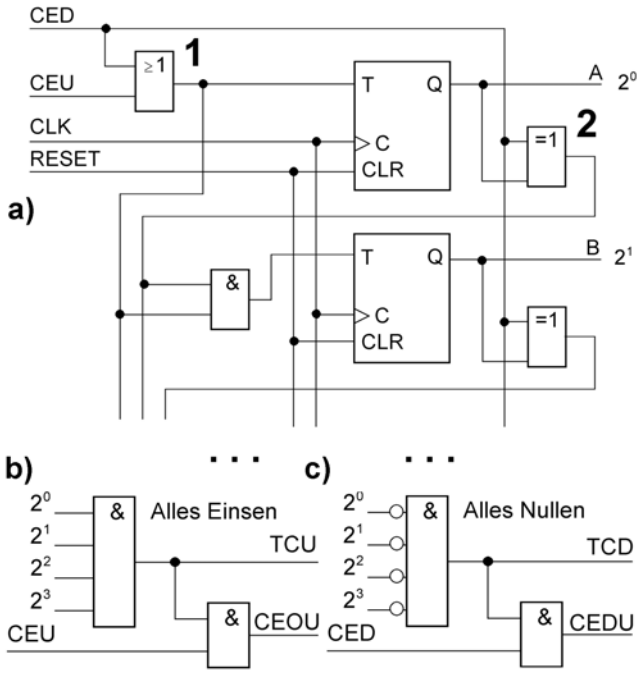

Abb. 6.37 Ausschnitt aus einem generischen Vorwärts-Rückwärts-Zähler. Den Zählnetzwerken des Vorwärtszählens werden beim Rückwärtszählen die invertierten Ausgangssignale der Flipflops zugeführt. CED = Erlaubnis zum Rückwärtszählen (Down); CEU = Erlaubnis zum Vorwärtszählen (Up); 1 - allgemeine Zählerlaubnis; 2 - gesteuerte Invertierung. a) die ersten beiden Zählflipflops; b) Endzustand und Zählerlaubnisausgang für Vorwärtszählen; c) Endzustand und Zählerlaubnisausgang für Rückwärtszählen.

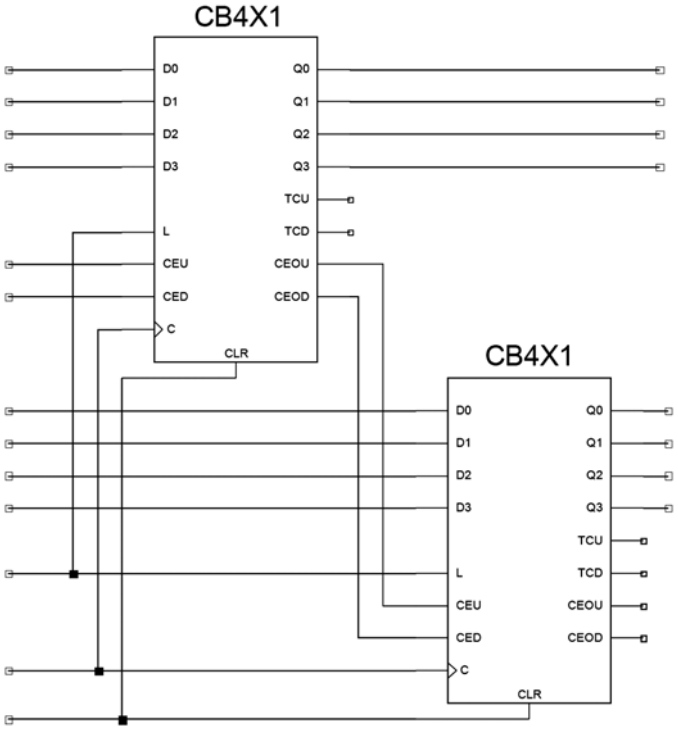

Abb. 6.38 Kaskadierung generischer Vorwärts-Rückwärts-Zähler. U = vorwärts (Up); R = Rückwärts (Down).

6.2.8 BCD-Zähler

Dezimalzähler zählen modulo 10 – also von 0 bis 9. Mehrstellige Dezimalzähler werden durch Kaskadieren solcher Funktionselemente aufgebaut. Soll die einzelne Dezimalstelle im 1-aus-n-Code dargestellt werden, kann man die Funktionselemente als Ring- oder Johnsonzähler auslegen. Oftmals braucht man aber die Dezimalstellen in binärer Codierung (BCD-Code von 0 = 0000B bis 9 = 1001B).

	2^3(D)	2^2 (C)	2^1 (B)	2^0 (A)	TD_V	TC_V	TB_V	TA_V	TD_R	TC_R	TB_R	TA_R
0	0	0	0	0				!	!			!
1	0	0	0	1			!	!				!
2	0	0	1	0				!			!	!
3	0	0	1	1		!	!	!				!
4	0	1	0	0				!		!	!	!
5	0	1	0	1			!	!				!
6	0	1	1	0				!			!	!
7	0	1	1	1	!	!	!	!				!
8	1	0	0	0				!	!	!	!	!
9	1	0	0	1	!			!				!

Tabelle 6.5 Zustandsfolgetabelle der BCD-Zählung. Daneben die Ansteuerbedingungen für T-Flipflops. Die Ausrufungszeichen kennzeichnen die Stellungen, in denen der T-Eingang desjeweiligen Flipflops mit einer Eins anzusteuern ist. V = vorwärts; R = rückwärts.

Aus der Tabelle können die folgenden Ansteuergleichungen abgelesen werden. Dabei kann man die Belegungen 10 bis 15 (AH...FH) als Don't-Care-Bedingungen ansetzen.

a) Vorwärtszählen:

$$2^0\colon\ TA_V = 1$$
$$2^1\colon\ TB_V = \overline{D} \cdot A$$
$$2^2\colon\ TC_V = B \cdot A$$
$$2^3\colon\ TD_V = B \cdot C \cdot A \vee D \cdot A$$

b) Rückwärtszählen:

$$2^0\text{: } TA_R = 1$$
$$2^1\text{: } TB_R = B \cdot \overline{A} \vee C \cdot \overline{A} \vee D \cdot \overline{A}$$
$$2^2\text{: } TC_R = C \cdot \overline{B} \cdot \overline{A} \vee D \cdot \overline{A}$$
$$2^3\text{: } TD_R = \overline{C} \cdot \overline{B} \cdot \overline{A}$$

$$TC\ (\text{vorwärts}) = D \cdot \overline{C} \cdot \overline{B} \cdot A;\ \text{vereinfacht } D \cdot A \quad (6.20)$$

Praxistipp: Der Dezimalzähler hat in der Vergangenheit eine beträchtliche Bedeutung gehabt, da Zählschaltungen oftmals direkt mit Bedien- und Anzeigemitteln verbunden waren (Rändelschalter, Siebensegmentanzeigen usw.). Heutzutage liegt es nahe, Bedienung und Anzeige programmtechnisch zu erledigen (Mikrocontroller). Zählschaltungen kommen nur dann zum Einsatz, wenn es aus Geschwindigkeitsgründen erforderlich ist. Dann dürfte es zumeist am besten sein, rein binär zu zählen (oder in einem beliebigen anderen Code gemäß dem jeweiligen Awendungsproblem), und das Wandeln ins Dezimale oder aus dem Dezimalen dem Programm zu überlassen.

6.2.9 Zählen mit beliebiger Zählweite

Um mit Binär- oder BCD-Zählern eine beliebige Zählweite ($\neq 2^n$ oder 10^n) zu verwirklichen, gibt es mehrere Möglichkeiten:

- die Zustandsfolgetabelle aufstellen und die Zählnetzwerke entsprechend entwerfen (Entwurf als Zustandsautomat),
- den Zähler mit RAM-Zuordnern aufbauen und entsprechende Bitmuster erzeugen (wird vom Entwicklungssystem erledigt),
- den letzten gewünschten Zählzustand (Endwert) decodieren und den Zähler in den anfänglichen Zählzustand zurückführen (z. B. durch Rücksetzen),
- den Zähler so voreinstellen (z. B. durch Laden), dass er nicht von Null an, sondern von einem Anfangswert an zählt.

Decodieren und rücksetzen

Der Zähler wird zurückgesetzt, wenn die jeweils gewünschte letzte Zählerstellung (Endwert) erreicht ist. Dann beginnt er erneut vom Anfangswert an zu zählen.

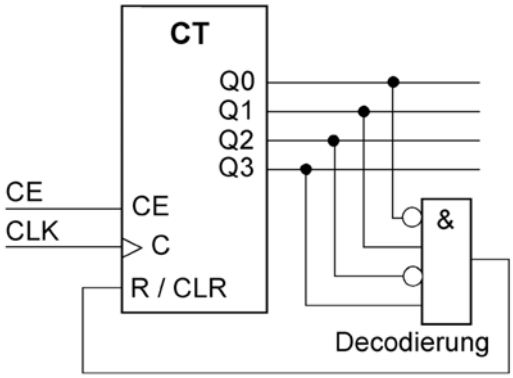

Abb. 6.39 Zählen mit beliebiger Zählweite (1). Decodieren und Rücksetzen (Grundschaltung). Hier wird die Zählerstellung 1010B = 10 decodiert. Beim synchronen Rücksetzen (R) zählt der Zähler 0 – 1 – ... 9 – 10 – 0, also modulo 11, beim asynchronen Löschen (CLR) hingegen nur modulo 10, weil beim Erreichen der Stellung 10 nicht erst mit dem folgenden Takt, sondern sofort zurückgesetzt wird.

Das Rücksetzen

Es kommt darauf an, ob synchron oder asynchron zurückgesetzt wird. Ist von Null an modulo z zu zählen, so ist folgender Zählerstand zu decodieren:

- Beim synchronen Rücksetzen: z – 1. Daraufhin wird mit dem nächsten Takt die Stellung Null eingenommen.
- Beim asynchronen Löschen: z. Diese Belegung wird kurzzeitig erreicht. Sie bewirkt, dass im gleichen Takt die Stellung Null erzwungen wird. Im Grunde entsteht am Decoderausgang nur ein Glitch, und die Funktion der Schaltung ist unsicher (da die Flipflops unterschiedliche Verzögerungszeiten beim Rücksetzen haben).

Das Decodieren eines Zählerstandes z ergibt beim synchronen Rücksetzen ein Zählen modulo z + 1, beim asynchronen modulo z.

Veränderliche Zählweite

Die einfache Decodierung eines Festwertes wird durch das Vergleichen des aktuellen Zählerstande mit einem ladbaren Endwert ersetzt.

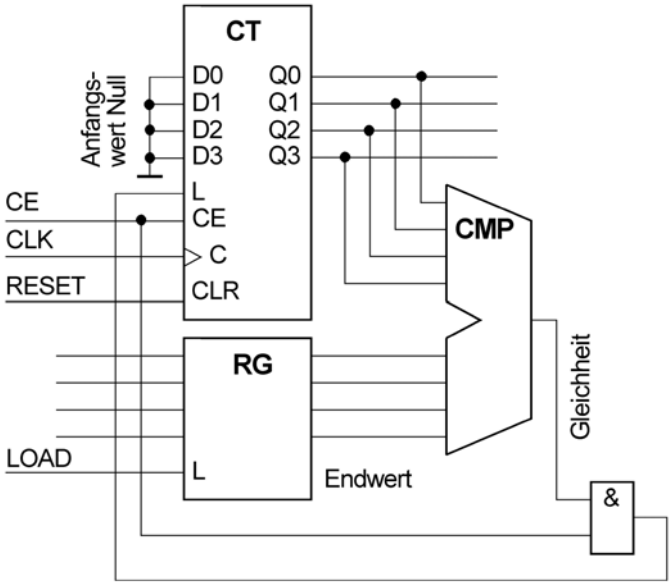

Abb. 6.40 Zählen mit beliebiger Zählweite (3). Die Zählweite kann während des Betriebs verändert werden. Der jeweilige Endwert ist in ein Register zu laden. Der Zähler wird durch Laden mit dem Festwert Null in den Anfangszustand zurückgeführt. Somit kann der Rücksetz- oder Löscheingang zum Anfangsrücksetzen genutzt werden.

Vom Anfangswert bis zum Ende zählen

Es wird von einem Anfangswert a an gezählt, bis der Zähler seinen gleichsam natürlichen letzten Zählzustand (Endwert) erreicht hat (binär 11...1B, dezimal 99...9). Dann wird der Zähler wieder mit dem Anfangswert geladen. Der Anfangswert a ergibt sich gemäß Tabelle 6.6.

Laden	Synchron	Asynchron
Binärzähler	$a = 2^n - z$ (Zweierkomplement)	$a = 2^n - z - 1$ (Einerkomplement)
Dezimalzähler	$a = 10^n - z$ (Zehnerkomplement)	$a = 10^n - z - 1$ (Neunerkomplement)

Tabelle 6.6 Anfangswerte beim Zählen mit beliebiger Zählweite.

Das Zählen vom Anfangswert an hat seinen Ursprung in dem Bestreben, Zählerschaltkreise mittleren Intergrationsgrades so gut wie möglich auszunutzen. Der typische Zählerschaltkreis ist (1) ladbar und er enthält (2) eine UND-Verknüpfung, die den jeweiligen Endwert decodiert. Zähler in programmierbaren Schaltkreisen werden hingegen immer von Grund auf synthetisiert. Dabei werden ungenutzte Gatter und Flipflops der Funktionselemente typischerweise weggelassen. Gatternetze werden auf ihre Booleschen Gleichungen zurückgeführt, die dann minimiert und mit den Ressourcen des Schaltkreises implementiert werden. Beispielsweise werden kaskadierte UND-Verknüpfungen mit einem einzigen UND-Gatter aufgebaut, sofern die Gatter in den Zellen genügend viele Eingänge haben. Es kann

sein, dass es weniger Aufwand erfordert, einen bestimmten Endwert zu decodieren und den Zähler zu löschen, als ihn ladbar auszulegen. Oftmals bleiben sich die tatsächlichen Aufwendungen gleich.

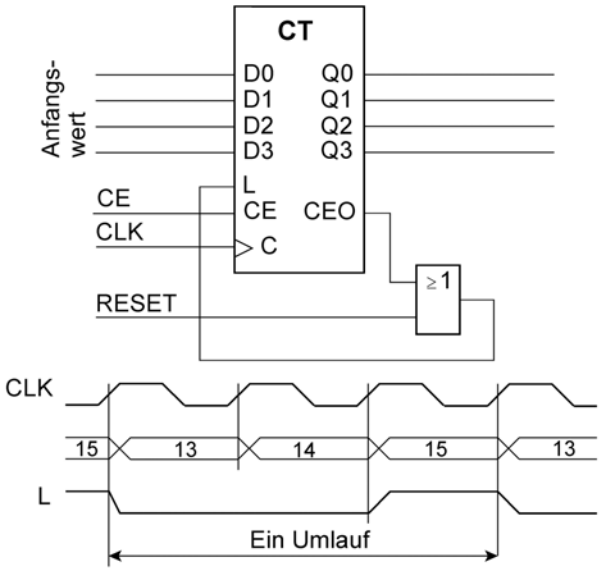

Abb. 6.41 Vom Anfangswert an zählen. Der Anfangswert ist das Zweier- oder Zehnerkomplement. Dieser vierstellige Binärzähler zählt modulo 3. Hierzu wird er mit dem Zweierkomplement von 3 synchron geladen (16 – 3 = 13). Die Zählfolge: 13 – 14 – 15 – 13 usw.).