1 Schaltungsentwurf mit Hardwarebeschreibungssprachen

1.1 Vom Schaltplan zur Schaltungssynthese

Digitaltechnik ist zum Schlagwort geworden. Wir leben im *digitalen Zeitalter*. Die Natur ist kontinuierlich. Wir *digitalisieren* ihre Erscheinungen. Für vieles besitzen wir Rechenmodelle mit Zahlen aus den binären Werten ‚0' und ‚1'. Nicht nur der PC führt digitale Operationen aus. Digital-Chips sind in Maschinen, Fahrzeugen, Haushaltsgeräten, im Handy, in der Chipkarte, ...

Die Anfänge der Digitaltechnik sind viel älter als die Elektronik. Als erste digitale Maschinen können wir die kleinen und großen mechanischen Apparate mit komplizierten Getrieben für die Abarbeitung der dezimalen Rechenoperationen aus den zurückliegenden Jahrhunderten ansehen. Bereits im 17. Jahrhundert baute Wilhelm Schickard eine mechanische Rechenmaschine für die vier Grundrechenarten.

Nach der Einführung der elektromechanischen Relais in den 20er Jahren des letzten Jahrhunderts konnten bereits anspruchsvolle Steuerungen entwickelt werden. Bild 1.1 zeigt an einem Beispiel, dass mit zwei Typen von Relais logische Verknüpfungen von binären Signalen ausgeführt werden können.

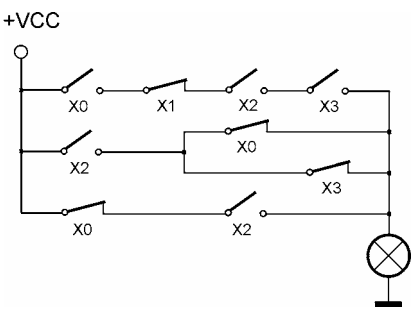

Bild 1.1 Schaltplan einer Relais-Schaltung

Dabei entspricht eine Reihenschaltung von Kontakten einer UND-Verknüpfung. Parallele Kontakte bilden eine ODER-Schaltung.

Die erste programmgesteuerte Rechenmaschine von Konrad Zuse aus dem Jahr 1941 enthielt ca. 2000 elektromechanische Relais.

Aus dieser Zeit stammen Theorien und Terminologien, die wir noch heute benutzen, wie **Schaltplan**, Schaltalgebra und Automatentheorie.

Auch die Elektronenröhre ist für eine kurze Zeit in digitalen Systemen eingesetzt worden. Das spektakulärste Beispiel ist der Computer ENIAC mit 20 000 Elektronenröhren aus dem Jahr 1946.

Aus heutiger Sicht hat erst die Entwicklung der Transistortechnik die Digitaltechnik eingeleitet. Die integrierten mikroelektronischen TTL-Schaltungen waren ab Anfang der 60er Jahre mit einem wachsenden Bausteinsortiment für digitale Grundoperationen verfügbar.

10 Jahre später brachte die MOS-Technik eine wesentliche Steigerung des Integrationsgrades. Eine der ersten Anwendungen der neuen Technik waren Taschenrechnerschaltkreise, die als Low-cost-Geräte die Dezimaloperationen ihrer mechanischen Vorfahren ausführen konnten.

Es folgten hoch- und höchstintegrierte Schaltungen für Mikroprozessoren, Speicher, Schnittstellen-Controller und applikationsspezifische Schaltungen. Heute werden Mikroprozessoren in Milliardenstückzahlen produziert. Davon geht nur ein kleiner Anteil in die PCs. Die Hauptanwendungen liegen in den kaum sichtbaren „Embedded Systemen“.

Für eine lange Zeit konnten spezielle Hardwarelösungen in großem Maße durch die Softwareentwicklung für universelle Mikroprozessorsteuerungen ersetzt werden. Damit beschränkte sich der Chip- und Hardware-Entwurf auf relativ wenige Spezialisten. Auch in der Hochschulausbildung wurde der Schwerpunkt auf die Software gelegt.

Inzwischen ist der Stellenwert der Hardware-Entwicklung wieder gewachsen. Durch die Verfügbarkeit programmierbarer Bauelemente werden Applikationsschaltungen verstärkt beim Anwender entwickelt.

Die Anfänge der programmierbaren Bausteine (programmable logic device, PLD) gehen auf die 80er Jahre zurück. Sie enthalten programmierbare Macrozellen, die zu anwenderspezifischen Schaltungen zusammengefügt werden. Mit einem einfachen PLD-Baustein können die Funktionen von mehreren Standardbausteinen zusammengefasst werden.

Heute sind hochintegrierte komplexe PLDs (CPLDs) mit Hunderten von Macrozellen verfügbar. Ein weiteres Architekturprinzip verwendet Felder programmierbarer Macrozellen und programmierbare Leitungssegmente in Form von beim Anwender programmierbaren Gatearrays (field programmable gate arrays, FPGAs). Damit stehen programmierbare Bausteine mit Hardware-Ressourcen im Bereich von Millionen Gatterfunktionen zur Verfügung.

Wenn inzwischen für fast alle ingenieurtechnischen Aufgaben Computerprogramme für die Entwurfsunterstützung genutzt werden, ist diese Rechnerunterstützung auch für den Entwurf der Computerhardware selbstverständlich.

Insbesondere von den Herstellern programmierbarer Bauelemente sind Entwurfswerkzeuge für digitale Schaltungen bereitgestellt worden.

Diese Entwicklungssysteme waren am Anfang auf die Schaltbild-Darstellung digitaler Schaltungen orientiert. Zusätzlich konnten Gleichungen und Zuordnungstabellen logischer Signale angegeben werden. Bestandteil dieser Systeme sind die Bibliotheken von Funktionsmodulen digitaler Grundschaltungen. Diese Entwurfsmodule sind den digitalen Standardbausteinen niederen und mittleren Integrationsgrades vergleichbar. Dabei benutzt jeder Hersteller und jede Architektur proprietäre Bibliotheken.

Der Entwurf großer digitaler Systeme muss vorgefertigte, aber flexible Lösungen komplexer Funktionsmodule benutzen. Die sich ständig vergrößernden Modulbibliotheken für die Schaltbildbeschreibung sind damit an ihre Grenzen gestoßen.

Stattdessen sind Hardwarebeschreibungssprachen (**H**ardware **D**escription **L**anguage, HDL) für die algorithmische Beschreibung der Funktionen und Strukturen von digitalen Systemen eingeführt worden. Damit lassen sich Funktionseinheiten sehr flexibel mit Text-Anweisungen formulieren. Das Entwicklungssystem muss diese Algorithmen in die Strukturen aus digitalen Grundschaltungen übersetzen.

Dabei hat die Software-Entwicklung das Vorbild für eine effektive Hardware-Entwicklung geliefert. Für die Programm-Entwicklung werden seit Jahren Programmgerüste eingesetzt, die vom Programmierer unter Nutzung von Objektbibliotheken modifiziert werden.

Die zur Zeit wichtigsten Hardwarebeschreibungssprachen sind VERILOG und VHDL (**V**ery high speed integrated circuit **H**ardware **D**escription **L**anguage).

Besonders in Europa ist VHDL in der Hochschulausbildung dominierend. Dazu haben die für Lehraufgaben kostenlos nutzbaren Entwicklungssysteme der FPGA-Hersteller entscheidend beigetragen. Diese Systeme können auch auf dem persönlichen PC installiert werden.

In VHDL wird das Verhalten und die Struktur von digitalen Systemen Hersteller- und weitgehend Baustein-unabhängig beschrieben. Schaltbild-Beschreibungen können in VHDL übertragen werden. Darüber hinaus werden die Aufgabenstellungen für den Schaltungsentwurf auf hohem Abstraktionsniveau mit leistungsfähigen Operationen, zum Beispiel für die Dual-Arithmetik und Zählfunktionen, formuliert. Diese Beschreibungen gehen weit über die Möglichkeiten der Schaltbild-Darstellung hinaus.

Die Beschreibung der Algorithmen nutzt die aus den höheren Programmiersprachen bekannten Anweisungen.

Die VHDL-Beschreibung wird in der **Schaltungs-Synthese** in digitale Grundschaltungen umgerechnet.

Auf der Ebene der VHDL-Beschreibung kann die Funktion durch **Simulation** überprüft werden. In einem zweiten Schritt müssen Konfigurationsdateien für die bausteinspezifischen Macrozellen der Zielarchitektur erzeugt werden. Dabei entstehen bausteinabhängige Netzlisten für die Zusammenschaltung der Macrozellen. In weiteren Verarbeitungsschritten erfolgen die lokale Zuordnung (Platzierung) und die interne Verdrahtung (Routing) auf dem Chip.

Erfahrene Spezialisten können große digitale Systeme mit diesen hochproduktiven Entwicklungssystemen mit geringem Zeitaufwand entwickeln.

Bild 1.2 zeigt das FPGA-Layout für einen vereinfachten 8-Bit-Prozessorkern mit „8051“-Architektur, der aus einer VHDL-Synthese-Beschreibung erzeugt worden ist.

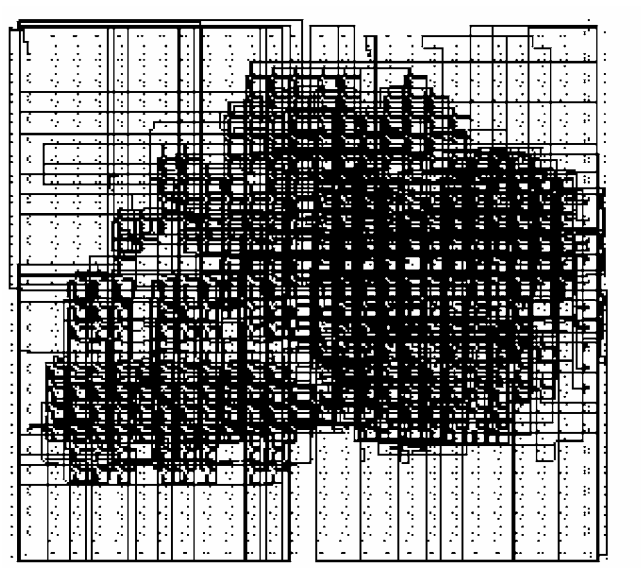

Bild 1.2 FPGA-Layout für Prozessorkern

Dabei wird ein Low-cost-FPGA-SPARTAN1 mit 30 000 Gatterfunktionen eingesetzt. Diese Anwendung ist durch die Nutzung von Ressourcen für interne Speicherblöcke für Programm- und Daten-Speicher möglich geworden.

Die Methoden des industriellen Schaltungsentwurfs müssen sich an Systemen mit einigen 100 000 (oder Millionen) Gatterfunktionen orientieren.

Mit der Standardisierung von VHDL existiert eine einheitliche Dokumentationsplattform für große digitale Systeme.

Teilweise liefern die Hersteller programmierbarer Bausteine kostenlose VHDL-Lösungen für oft gebrauchte Funktionseinheiten. Lösungen für Teilsysteme werden allerdings oft auch in Form von Netzlisten für bestimmte Bausteinfamilien angeboten. Diese besitzen ein VHDL-Interface für den Einsatz in weiteren VHDL-Entwürfen. Die Quellen sind dann aber geschützt und nicht modifizierbar.

Ein typischer Bestandteil von Entwicklungssystemen sind sogenannte CORE-Generatoren, mit denen komplexe digitale Baugruppen konfiguriert werden können. Diese liefern ebenfalls in VHDL-Entwürfe einbindbare Netzlisten.

Wie in der Software-Entwicklung entstehen VHDL-Bibliotheken für digitale Funktionseinheiten. Systementwürfe für große digitale Systeme werden als IP-cores (intellectual property, „geistiges Eigentum“ an Systementwürfen) kommerziell vertrieben.

Euphorische Prognosen, dass in Zukunft mit VHDL ein Hardwareentwurf ohne detaillierte Hardwarekenntnisse möglich sein soll, sind jedoch übertrieben und gefährlich. Durch eine „hardwareferne“ Verhaltensbeschreibung werden Anfänger zur empirischen Arbeitsweise verleitet.

In den folgenden Abschnitten werden die Grundlagen der digitalen Schaltungen zusammen mit der Einführung in eine “hardwarenahe“ VHDL-Beschreibung dargestellt. Beim VHDL-Entwurf muss das Ergebnis der Schaltungssynthese im Prinzip vorhersehbar sein.

Die Transparenz zwischen VHDL-Beschreibungen und den daraus entstehenden Schaltungsstrukturen ist unverzichtbar. Das gilt für die Umsetzung von Schaltbildern in VHDL-Beschreibungen und für die Zuordnung der Schaltungsstrukturen zu VHDL-Modellen. Deshalb werden in den folgenden Kapiteln für die wichtigsten Grundschaltungen der Digitaltechnik die zugehörigen VHDL-Beschreibungen entwickelt.

Auch der Entwurf digitaler Systeme, die nicht mit programmierbaren Bauelementen realisiert werden sollen, benutzen diese Entwicklungswerkzeuge. Es ist naheliegend, auch für diese Anwendungen den Entwurf eines neuen Systems mit einem FPGA oder der Zusammenschaltung von vielen FPGAs zu testen. Daran kann sich eine ASIC-Lösung mit fester Maskenprogrammierung anschließen.

Mit der Verfügbarkeit immer höher integrierter programmierbarer Bauelemente verlagern sich die Aufgaben des Schaltungsentwurfs auf immer größere Systeme.

Für den rechnergestützten Entwurf werden bisher geltende Optimierungskriterien durch neue Entwurfsregeln ersetzt.

- Routineprozesse der Schaltungsreduzierung werden vom Computer übernommen.
- Eine Minimierung der Anzahl von Speicherelementen ist oft nicht mehr notwendig.
- Zusätzliche Speicherelemente können für die Erhöhung der Arbeitsgeschwindigkeit eingesetzt werden.
- In vielen Fällen ist die Minimierung der Anzahl von Eingangssignalen für logische Teilschaltungen wichtig.
- Der Einsatz von asynchronen Schaltungen besitzt keine praktische Bedeutung mehr.
- Die Realisierung synchroner sequenzieller Schaltungen ist vorwiegend auf taktflankengesteuerte D-Flipflops orientiert.
- Der Entwurf großer Systeme erfordert leistungsfähige Entwurfselemente.
- Entwurfselemente können durch hierarchische Strukturen beschrieben werden.
- Neue Entwurfselemente, z. B. verteilte Speicherblöcke, Schieberegistermodule oder Multipliziermatrizen, ermöglichen neuartige Lösungen für digitale Systeme.

1.2 Logische Operationen in digitalen Schaltungen

Digitale Schaltungen arbeiten mit binären Signalen. Mit binären Signalen können logische Operationen ausgeführt werden. Logiksysteme mit mehrwertigen Signalen werden nur in Sonderfällen eingesetzt.

Binäre Signale werden durch Spannungen oder Ströme definierter Werte physikalisch realisiert. Die Grundlage der digitalen Informationsverarbeitung in hochintegrierten Bauelementen sind heute fast ausschließlich integrierte CMOS-Schaltungen. Den logischen Signalen ‚1' und ‚0' entsprechen Signalpegel mit Spannungen von 5 V bis unter 2 V und 0 V.

Für diese CMOS-Schaltungen existieren Schaltungsrealisierungen für die digitalen Grundoperationen. Diese enthalten paarweise zusammengeschaltete komplementä-

re n- und p-Kanal-MOS-Transistoren, die von den Eingangssignalen gegensinnig geschaltet werden. Bild 1.3 zeigt die Grundschaltungen für die logischen Grundfunktionen NAND und NOR.

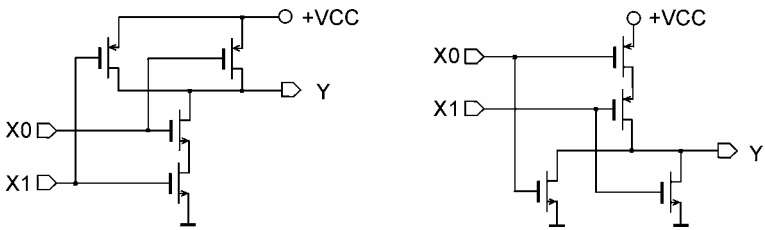

Bild 1.3 CMOS-Schaltungen für die NAND- und die NOR-Verknüpfung

Der Entwurf größerer digitaler Schaltungen führt immer auf eine Zusammenschaltung logischer Grundschaltungen. Damit verlagert sich der Entwurf digitaler Schaltungen auf die Zusammenschaltung logischer Grundelemente.

Ein Logikelement mit zwei Eingängen kann formal eine von 16 möglichen logischen Funktionen beschreiben. Tabelle 1.1 ordnet den 4 Eingangssignalkombinationen 16 verschiedene Ausgangssignalkombinationen zu. Für die Ein- und Ausgänge werden die Signalwerte ‚0' und ‚1' zugelassen. Theoretisch ergeben sich für n Eingangsvariable 2^{2^n} Funktionen. Mit dem Übergang auf 3 Eingangsvariable kommen wir auf 256, mit 4 Eingangsvariablen auf 65 536 mögliche logische Funktionen.

Neben einigen Trivialfällen existieren mehrere technisch sinnvolle logische Operationen.

Tab. 1.1 Logische Funktionen für zwei Eingangsvariable

X1	X0	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7	Y8	Y9	YA	YB	YC	YD	YE	YF
0	0	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
0	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	0	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
		'0'	NOR		NOT X1		NOT X0	XOR	NAND	AND	XNOR	X0		X1		OR	'1'

In Abschnitt 2.1 sehen wir, dass sich alle kombinatorischen digitalen Schaltungen mit wenigen Grundoperationen beschreiben lassen. Dafür reichen die Konjunktion (AND-, UND-Verknüpfung) und Disjunktion (OR, ODER-Verknüpfung) in Verbindung mit der Negation (unäre NOT-Operation, Inversion) aus.

Der praktische Schaltungsentwurf vereinfacht sich jedoch, wenn wir eine größere Anzahl von „Grundoperationen“ in Form von Entwurfselementen zulassen.

Bereits die Erweiterung auf XOR (exclusive or, Antivalenz) und XNOR (negiertes exclusive or, Äquivalenz) erweitert diese Möglichkeiten beträchtlich. Zusätzlich werden die Gatter auf mehrere Eingänge erweitert. Außerdem soll die Negation der Eingangssignale in modifizierten Gattersymbolen berücksichtigt werden.

Für die Schaltungssymbole logischer Grundschaltungen existieren mehrere Standards. Viele deutschsprachige Lehrbücher benutzen die Symbole der DIN-Vorschrift 40900. Diese DIN-Vorschrift lässt aber auch die Anwendung der international üblichen Symbole zu. Die letzteren werden ausschließlich bei der Schaltbildeingabe in den international verbreiteten Entwurfssystemen des Schaltungsentwurfs genutzt, auf die wir im weiteren Bezug nehmen müssen. Außerdem sind sie übersichtlicher. Wir werden deshalb nur die internationalen Symbole verwenden.

Bild 1.4 zeigt die Entsprechungen der unterschiedlichen Symbole.

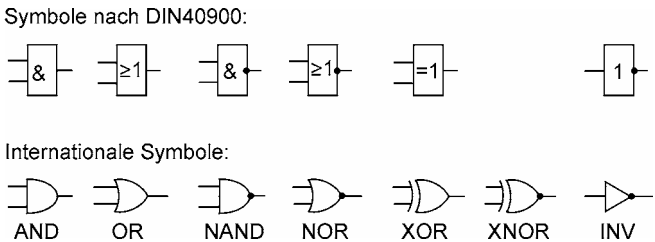

Bild 1.4 Schaltsymbole für logische Grundoperationen

Auch für die Angabe der Operationen in Gleichungen sind unterschiedliche Zeichen in Gebrauch. Wir verwenden die Symbole $\vee$ und $\wedge$ für die OR- und AND-Operation. Das Operatorsymbol für die AND-Verknüpfung kann auch entfallen. Davon machen wir später Gebrauch. Die Negation von Operanden wird durch Überstreichung oder ein vorangestelltes ! angegeben. Für die XOR-Funktion ist das +-Zeichen wegen der später aufgezeigten Beziehung zur binären Addition sinnvoll.

Die Regeln der Schaltalgebra beziehen sich auf die AND- und OR-Operation:

$$A \wedge A = A \qquad A \vee A = A$$
$$A \wedge \overline{A} = '0' \qquad A \vee \overline{A} = '1'$$
$$A \wedge '1' = A \qquad A \vee '1' = '1'$$
$$A \wedge '0' = '0' \qquad A \vee '0' = A$$

Ein erstes Distributivgesetz enthält die Vorschrift für das Ausklammern einer in mehreren disjunktiv verknüpften Konjunktionen auftretenden Variablen.

$(A \wedge B) \vee (A \wedge C) = A \wedge (B \vee C)$

Ein wichtiger Sonderfall dieser Regel

$(A \wedge B) \vee (A \wedge \overline{B}) = A \wedge (B \vee \overline{B}) = A$

dient der manuellen Vereinfachung logischer Ausdrücke. Ein zweites Distributivgesetz bezieht sich auf das Ausklammern von Variablen aus konjunktiv verknüpften Disjunktionen.

$(A \vee B) \wedge (A \wedge C) = A \vee (B \wedge C)$

Aus diesen Regeln lassen sich die DeMorganschen Regeln ableiten:

$\overline{A \wedge B \wedge C} = \overline{A} \vee \overline{B} \vee \overline{C}$

$\overline{A \vee B \vee C} = \overline{A} \wedge \overline{B} \wedge \overline{C}$

Diese lassen sich schaltungstechnisch interpretieren:

> Ein NAND-Element entspricht einem OR-Element mit negierten Eingangssignalen. Ein NOR-Element entspricht einem AND-Element mit negierten Eingangssignalen.

Zusätzlich benötigen wir für die XOR-Operation die folgenden Beziehungen:

$A + A = '0'$

$A + \overline{A} = '1'$

$\overline{A} + B = \overline{A + B}$

Die letzte Beziehung mit der trivialen Deutung:

> Die Negation der Antivalenz ist die Äquivalenz.

ergibt die nützliche Relation:

$A \wedge \overline{B} \vee \overline{A} \wedge B = \overline{\mathrm{A} \wedge \mathrm{B} \vee \overline{\mathrm{A}} \wedge \overline{\mathrm{B}}}$

Jeder logische Ausdruck kann als Disjunktion von Konjunktionen (Produktterme) oder als konjunktive Verknüpfung von Disjunktionen ausgedrückt werden. In der Technik wird die erste Form bevorzugt. Darauf kommen wir im Abschnitt 2.1 zurück.

Jeder Schaltung aus logischen Grundelementen entspricht eine Gleichung mit den entsprechenden logischen Operatoren. Umgekehrt können wir jede logische Gleichung als Schaltbild darstellen.

Als Beispiel wählen wir eine Schaltung nach Bild 1.5, mit der alle 16 logischen Funktionen für zwei Eingangsvariable a und b mit den Steuersignalen s3, s2, s1 und s0 eingestellt werden sollen.

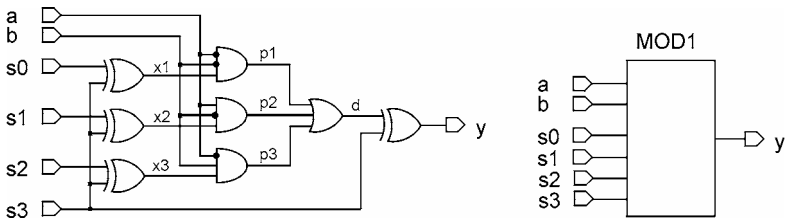

Bild 1.5 Schaltung aus der Zusammenschaltung von Grundelementen

Die Schaltung interpretieren wir als Bildungsvorschrift für die folgende Gleichung.

$$Y=S3+((\overline{A} \wedge \overline{B} \wedge (S0 + S3)) \vee (A \wedge \overline{B} \wedge (S1+S3)) \vee (\overline{A} \wedge B \wedge (S2+S3)))$$

Die Schaltung kann als Funktionselement in größeren Systemen eingesetzt werden.

1.3 Dualarithmetik

Das Wort *digital* weist darauf hin, dass in der Digitaltechnik neben den logischen Signalen vor allem *Zahlen* verarbeitet werden.

Weil der Mensch 10 Finger hat, rechnen wir im **Dezimalsystem** mit der Basis 10. Dabei werden Zahlen aus Koeffizienten für Zehnerpotenzen gebildet.

$$Z = d_{n-1} * 10^{n-1} + d_{n-2} * 10^{n-2} + ... + d_2 * 10^2 + d_1 * 10 + d_0 * 10^0$$

Andere Zahlensysteme mit jeder beliebigen Basis sind möglich. Für digitale Schaltungen mit binären Signalen ist das **Dualsystem** besonders geeignet.

Im Dualsystem werden positive ganze Zahlen in der Form

$$Z = b_{n-1} * 2^{n-1} + b_{n-2} * 2^{n-2} + ... + b_2 * 2^2 + b_1 * 2 + b_0 * 2^0$$

dargestellt. Die Koeffizienten b_i mit den Werten '0' und '1' repräsentieren die Zahl im Dualcode. Zum Beispiel kann für die Dezimalzahl 177 die folgende Dualdarstellung angegeben werden:

$$Z = 1 * 2^7 + 0 * 2^6 + 1 * 2^5 + 1 * 2^4 + 0 * 2^3 + 0 * 2^2 + 0 * 2^1 + 1 * 2^0 = 10110001$$

Die **duale Addition** wird durch die stellenweise binäre Addition mit Übertrag ausgeführt.