

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV RADIOELEKTRONIKY

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF RADIO ELECTRONICS

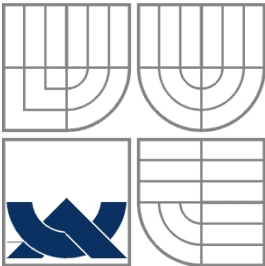
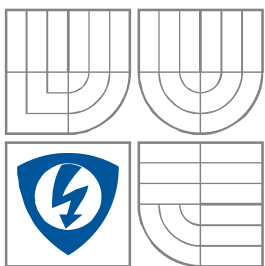
MULTIMEDÁLNÍ JAZYKOVÁ PODPORA DSP V NĚMČINĚ

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

PAVEL ŠKŮREK

BRNO 2008

	VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ BRNO UNIVERSITY OF TECHNOLOGY
	FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ ÚSTAV RADIOELEKTRONIKY FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION DEPARTMENT OF RADIO ELECTRONICS

MULTIMEDÁLNÍ JAZYKOVÁ PODPORA DSP V NĚMČINĚ

MULTIMEDIA LANGUAGE SUPPORT OF DSP IN GERMAN

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

Pavel Škůrek

VEDOUCÍ PRÁCE
SUPERVISOR

prof. Ing. Milan Sigmund, CSc.

BRNO, 2008

LICENČNÍ SMLOUVA

POSKYTOVANÁ K VÝKONU PRÁVA UŽÍT ŠKOLNÍ DÍLO

uzavřená mezi smluvními stranami:

1. Pan/paní

Jméno a příjmení: Pavel Škůrek
Bytem: Vilémov 113, 783 22
Narozen/a (datum a místo): 5. ledna 1986 ve Šternberku

(dále jen „autor“)

a

2. Vysoké učení technické v Brně

Fakulta elektrotechniky a komunikačních technologií
se sídlem Údolní 53, Brno, 602 00
jejímž jménem jedná na základě písemného pověření děkanem fakulty:
prof. Dr. Ing. Zbyněk Raida, předseda rady oboru Elektronika a sdělovací technika
(dále jen „nabyvatel“)

Čl. 1

Specifikace školního díla

1. Předmětem této smlouvy je vysokoškolská kvalifikační práce (VŠKP):

- ☐ disertační práce
- ☐ diplomová práce
- ☒ bakalářská práce
- ☐ jiná práce, jejíž druh je specifikován jako
(dále jen VŠKP nebo dílo)

Název VŠKP: Multimediální jazyková podpora DSP v němčině

Vedoucí/ školitel VŠKP: prof. Ing. Milan Sigmund, CSc.

Ústav: Ústav radioelektroniky

Datum obhajoby VŠKP: _____

VŠKP odevzdal autor nabyvateli*:

- ☒ v tištěné formě – počet exemplářů: 2
- ☒ v elektronické formě – počet exemplářů: 2

2. Autor prohlašuje, že vytvořil samostatnou vlastní tvůrčí činností dílo shora popsané a specifikované. Autor dále prohlašuje, že při zpracovávání díla se sám nedostal do rozporu s autorským zákonem a předpisy souvisejícími a že je dílo dílem původním.
3. Dílo je chráněno jako dílo dle autorského zákona v platném znění.
4. Autor potvrzuje, že listinná a elektronická verze díla je identická.

* hodící se zaškrtněte

Článek 2

Udělení licenčního oprávnění

1. Autor touto smlouvou poskytuje nabyvateli oprávnění (licenci) k výkonu práva uvedené dílo nevýdělečně užít, archivovat a zpřístupnit ke studijním, výukovým a výzkumným účelům včetně pořizování výpisů, opisů a rozmnoženin.
2. Licence je poskytována celosvětově, pro celou dobu trvání autorských a majetkových práv k dílu.
3. Autor souhlasí se zveřejněním díla v databázi přístupné v mezinárodní síti
 - ☒ ihned po uzavření této smlouvy
 - ☐ 1 rok po uzavření této smlouvy
 - ☐ 3 roky po uzavření této smlouvy
 - ☐ 5 let po uzavření této smlouvy
 - ☐ 10 let po uzavření této smlouvy
(z důvodu utajení v něm obsažených informací)
4. Nevýdělečné zveřejňování díla nabyvatelem v souladu s ustanovením § 47b zákona č. 111/ 1998 Sb., v platném znění, nevyžaduje licenci a nabyvatel je k němu povinen a oprávněn ze zákona.

Článek 3

Závěrečná ustanovení

1. Smlouva je sepsána ve třech vyhotoveních s platností originálu, přičemž po jednom vyhotovení obdrží autor a nabyvatel, další vyhotovení je vloženo do VŠKP.
2. Vztahy mezi smluvními stranami vzniklé a neupravené touto smlouvou se řídí autorským zákonem, občanským zákoníkem, vysokoškolským zákonem, zákonem o archivnictví, v platném znění a popř. dalšími právními předpisy.
3. Licenční smlouva byla uzavřena na základě svobodné a pravé vůle smluvních stran, s plným porozuměním jejímu textu i důsledkům, nikoliv v tísní a za nápadně nevýhodných podmínek.
4. Licenční smlouva nabývá platnosti a účinnosti dnem jejího podpisu oběma smluvními stranami.

V Brně dne: 6. června 2008

.....
Nabyvatel

.....
Autor

Prohlášení

Prohlašuji, že svou bakalářskou práci na téma Multimediální jazyková podpora DSP v němčině jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením této bakalářské práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.

V Brně dne 6. června 2008

.....
podpis autora

Poděkování

Děkuji vedoucímu bakalářské práce Prof. Ing. Milanu Sigmundovi, CSc. za účinnou metodickou, pedagogickou a odbornou pomoc a další cenné rady při zpracování mé bakalářské práce.

V Brně dne 6. června 2008

.....
podpis autora

Abstrakt

Cílem práce je poskytnout německy hovořícímu studentovi jednoduchý a přehledný seznam pojmů, které se vyskytují v oblasti matematiky, fyziky a číslicového zpracování signálu. Jednotlivé pojmy nejsou nijak složitě vysvětlovány. Většinou je uvedena jen základní definice, co daný výraz znamená, popř. jeho funkce. To znamená, že tuto práci nemůžeme považovat za učebnici, jde pouze o takovou první pomoc pro tápajícího studenta.

Práce je rozdělena na 3 databáze. První databáze Mathematik obsahuje výčet všech matematických pojmů a vztahů na úrovni střední školy. Databáze obsahuje tyto kapitoly: Grundbegriffe, Mathematische Logik, Funktion, Trigonometrie des rechtwinkligen Dreiecks, kongruente Darstellung im Ebene, ähnliche Darstellung im Ebene, Trigonometrie, komplexen Zahlen, Logarithmus, Vektor, analytische Geometrie, Folge, Differenzrechnung und Integralgleichung, Matrizen, Determinante, Kombinatorik a Wahrscheinlichkeit. Druhá databáze Physik obsahuje základní fyzikální veličiny, které by měl student znát již před nástupem na vysokou školu. Databáze obsahuje tyto kapitoly: Kinematik, Dynamik, Mechanik der starren Körper, Mechanik der Flüssigkeit, Elektrizität, Magnetismus, Optik a Thermodynamik. Poslední databáze DSP je rozdělena na 4 části, rozdělené podle funkčních bloků každého převodníku a vysvětlují jednotlivé výrazy. Jedná se o Antialiasing, Analog-Digital Wandler, Digital-Analog-Wandler a Spektral Analyse. U této databáze je také přehledný česko-německý a německo-český slovník.

Tyto 3 databáze jsou zpracovány programem C++ Builder. Uživatel si každou databázi může otevřít zvlášť. Všechny databáze obsahují funkce jako například vyhledávání, procházení termínů pomocí stromové struktury, příbuzná témata atd. V poslední databázi může uživatel dokonce i přidávat a upravovat svoje vlastní termíny.

Klíčová slova:

Matematika
Fyzika
DSP
Antialiasing
Analogově digitální převodník
Digitálně analogový převodník
Spektrální analýza

Abstract

The aim of the work is to provide a German speaking student with a simple and well-arranged list of terms, that appear in the sphere of mathematics, physics and the digital signal processing. Each terms are not explained in a difficult way. Mostly, there is only the basic definition, what the term means, or his function. This mean that this work can't be taken as a textbook, but only a first aid for a funbling student.

The work id divided into 3 databases. The first database Mathematik contains the list of all mathematical terms and relations at the level of secondary school. The database contains these chapters: Grundbegriffe, Mathematische Logik, Funktion, Trigonometrie des rechtwinkligen Dreiecks, kongruente Darstellung im Ebene, ähnliche Darstellung im Ebene, Trigonometrie, komplexen Zahlen, Logarithmus, Vektor, analytische Geometrie, Folge, Differenzrechnung und Integralgleichung, Matrizen, Determinante, Kombinatorik a Wahrscheinlichkeit. The second database Physik contains the basic physical quantities, the student should know before entering the university. The database contains these chapters: Kinematik, Dynamik, Mechanik der starren Körper, Mechanik der Flüssigkeit, Elektrizität,

Magnetismus, Optik and Thermodynamik. The last database DSP is divided into 4 parts. It is divided by functional blocks of each transmission and it is explaining each terms. They are Antialiasing, Analog-digital Wandler, Digital-analog Wandler and Spektral analysis. By this database, there is well-arranged czech - german and german - czech dictionary.

These 3 databases are worked by the programm C++ Builder. The user can open each database separately. All databases contain functions like for example searching, viewing the terms by tree structure, similar themes and so on. In the last database the user can add and change his own terms.

Keywords:

- Mathematics
- Physikal science
- DSP
- Antialiasing
- Analog to digital convertor
- Digital to analog convertor
- Spektral analyse

Bibliografická citace

ŠKŮREK, P. *Multimediální jazyková podpora DSP v němčině*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2008. 137 s. Vedoucí bakalářské práce prof. Ing. Milan Sigmund, CSc., 1 příl.

Obsah

Obsah	7
Seznam vzorců	8
Seznam tabulek	10
1. Úvod	11
2. Deutsches Alphabet	12
3. Griechisches Alphabet	14
4. Mathematik	16
4.1 Grundbegriffe	16
4.2 Mathematische Logik	17
4.3 Funktion	19
4.4 Trigonometrie des rechtwinkligen Dreiecks	20
4.5 kongruente Darstellung im Ebene	20
4.6 ähnliche Darstellung im Ebene	21
4.7 Trigonometrie	22
4.8 komplexen Zahlen	22
4.9 Logarithmus	24
4.10 Vektor	24
4.11 analytische Geometrie	25
4.11.1 Abweichung der Gebilde in den Raum	26
4.11.2 Kreislinie	27
4.11.3 Ellipse	27
4.12 Folge	28
4.12.1 arithmetische Folge	29
4.12.2 geometrische Folge	29
4.12.3 Grenzwert der Folge	30
4.13 Differenzrechnung und Integralrechnung	30
4.14 Matrizen	32
4.14.1 Operation mit der Matrizen	33
4.15 Determinante	33
4.16 Kombinatorik	34
4.17 Wahrscheinlichkeit	35
5. Physik	37
5.1 Kinematik	37
5.2 Dynamik	38
5.2.1 Energie	40
5.3 Mechanik der starken Körper	41
5.4 Mechanik der Flüssigkeit	41
5.5 Elektrizität	42
5.6 Magnetismus	44
5.7 Optik	46
5.7.1 Brechungsgesetz	46
5.7.2 Wellenwesen der Leuchte	47
5.7.3 Abbildung in dem Kugelspiegel	48
5.8 Thermodynamik	48
6 DSP	51
6.1 Deutsch-Tschechisch Wörterbuch für DSP	51
6.1.1 Antialiasing	51
6.1.2 Abtastung	51
6.1.3 Quantisierung	52
6.1.4 Analog-Digital-Wandler	52
6.1.5 Digital-Analog-Wandler	53
6.1.6 Spektralanalyse	53
6.2 Tschechisch-Deutsch Wörterbuch	54
6.2.1 Antialiasing	54
6.2.2 Vzorkování	54
6.2.3 Kvantování	55
6.2.4 Analogově digitální převodníky	55

6.2.5 Digitálně analogové převodníky	56
6.2.6 Spektrální analýza	56
6.3 Antialiasing	57
6.4 Analog-Digital-Wandler	70
6.4.1 Abtastung	70
6.4.2 Quantisierung	73
6.4.3 Analog-Digital-Wandler	77
6.5 Digital-Analog-Wandler	96
6.6 Spektralanalyse	105
7 Rozvaha nad softwarovým řešením slovníku	107
8. Vytvoření programu	108
8.1 Nejdůležitější funkce programu	108
8.1.1 Spuštění programu	108
8.1.3 Funkce tlačítek vpřed a zpět	108
8.1.5 Funkce přidání vlastní poznámky	109
8.1.9 Funkce přepínání mezi databázemi	109
8.1.10 Funkce přidávání záznamu v databázi DSP	109
8.1.11 Funkce správa přidáných záznamů	111
8.2 Rozbor a vysvětlení nejdůležitějších částí zdrojového kódu	112
8.2.1 Definice globálních proměnných v databázi DSP	113
8.3 Vysvětlení funkcí, které jsou v programu volány	115
8.3.1 Funkce adding() pro přidání nového záznamu t termin	115
8.3.2 Funkce show(), která říká, jak zobrazovat detaily jednotlivých termínů	116
8.3.3 Funkce show1(), která říká, jak zobrazovat německo-český slovník	116
8.3.4 Funkce show2(), která říká, jak zobrazovat česko-německý slovník	116
8.3.6 Funkce termíny(), která označuje termíny databáze v textu	118
8.3.7 Funkce AddNode(), která přidává záznam do stromové struktury	119
8.3.8 Funkce uloz() pro uložení přidáných záznamů do textového souboru	119
8.3.8 Funkce reset(), která „čistí“ komponenty pro zobrazení dalšího záznamu	120
8.3.9 Funkce zobraz() pro zobrazení vybraných termínů programem	120
8.3.10 Funkce nacti(), která načte do databáze přidané termíny	123
8.3.11 Funkce zobraz2(), která zobrazuje termín na formuláři pro správu termínů	124
8.3.12 Funkce abeceda() pro zobrazení řecké abecedy	126
8.4 Ošetření funkce hlavních událostí programu	126
8.4.1 Události na úvodním dialogovém okně	126
8.4.2 Události na dialogových formulářích jednotlivých databází	126
8.5 Nedostatky programu	144
9 Závěr	144
10 Literatura	145

Seznam vzorců

4.4.1 Funktion Sinus	20
4.4.2 Funktion Cosinus	20
4.4.3 Funktion Tangens	20
4.4.4 Funktion Kotangens	20
4.6.1 Euklidischer Lehrsatz über die Höhe	22
4.6.2 Euklidischer Lehrsatz über die Kathete	22
4.7.1 Sinus Lehrsatz	22
4.7.2 Cosinus Lehrsatz	22
4.8.1 Form der komplexen Zahlen	23
4.8.2 Absolutwert der komplexen Zahl	23
4.8.3 Argument der komplex Zahl	23
4.8.4 exponentiale Form der komplexen Zahl	24
4.8.5 goniometrische Form der komplexen Zahlen	24
4.9.1 Logarithmus	24
4.10.1 Größe der Vektors	25
4.10.2 inneres Produkt der Vektors	25
4.10.3 vektorielles Produkt der Vectors	25
4.11.1 Parametergleichung der Linie in der Ebene	25
4.11.2 Allgemeine Gleichung der Linie	26

4.11.3	Parametergleichung der Ebene	26
4.11.4	Allgemeine Gleichung der Ebene	26
4.11.5	Abweichung von zwei Linien	26
4.11.6	Abweichung zweier Ebenen	26
4.11.7	Abweichung der Linie und der Ebene	27
4.11.8	Zentrale Gleichung der Kreislinie	27
4.11.9	Allgemeine Gleichung der Kreislinie	27
4.11.10	Zentrale Gleichung der Ellipse	27
4.11.11	Allgemeine Gleichung der Elipse	27
4.11.12	Zentrale Gleichung der Hyperbel	28
4.11.13	Allgemeine Gleichung der Hyperbol	28
4.11.14	Scheitelgleichung der Parabel	28
4.11.15	Allgemeine Gleichung der Parabel	28
4.12.1	arithmetische Folge	29
4.12.2	Summe der arithmetischen Folge	29
4.12.3	geometrische Folge	29
4.12.4	Summe der geomertischen Folge	29
4.13.1	Derivation der Funtion	31
4.13.2	uneigentliches Integral	31
4.15.1	Laplacetransformierte	33
4.15.2	Cramersch Regel	34
4.16.1	Variation	34
4.16.2	Variation mit der Rekursion	34
4.16.3	Permutation	34
4.16.4	Kombination	35
4.16.5	binomischer Lehrsatz	35
4.17.1	Wahrscheinlichkeit des zufälligen Ereignises	35
5.1.1	Geschwindigkeit	36
5.1.2	Kontinuerlich beschleunigte Bewegung	36
5.1.3	Bahn der kontinuierlich beschleunigte Bewegung	37
5.1.4	Freifall	37
5.1.5	Rundbewegung	37
5.2.1	Zweites Trägheitgesetz	38
5.2.2	Beweglichkeit des Körper	38
5.2.3	Arbeit	38
5.2.4	Leistung	39
5.2.5	Wirksamkeit	39
5.2.6	Potenzielle Energie	39
5.2.7	kinetische Energie	39
5.3.1	Kraftmoment	40
5.3.2	Druck	40
5.4.1	hydrostatischer Druck	40
5.5.1	elektrisches Feld	41
5.5.2	elektrische Spannung	41
5.5.3	Kapazität	41
5.5.4	Kapazität aus den Proportionen des Kondensators	42
5.5.6	Coulombsches Gesetz	42
5.5.7	elektrischer Strom	42
5.5.8	Ohm Gesetz	42
5.5.9	Widerstand aus den Proportionen des Drahtes	43
5.5.10	Arbeit des elektrischen Stroms	43
5.5.11	Leistung des elektrischen Stroms	43
5.6.1	Intensität des Magnetischen Feldes	44
5.6.2	Magnetische Flussdichte	44
5.6.3	Selbstinduktivität der Spule	45
5.6.4	Selbstinduktivität aus der Proportion der Spule	45
5.7.1	Wellenlänge	45
5.7.2	Widerschein	45
5.7.3	Brechungsgesetz	46
5.7.4	Abbildungsgleichung	47

5.8.1 Erster Hauptsatz.....	47
5.8.2 Zweiter Hauptsatz.....	48
5.8.3 Wärme.....	48
5.8.4 Schmelzwärme.....	48
6.3.1 Bandbreite.....	57
6.3.2 Differentialgleichung.....	60
6.3.3 diskrete Fourier-Transformation.....	60
6.3.4 Fourier-Transformation.....	62
6.3.5 Nyquist-Shannonsche Abtasttheorem.....	65
6.3.6 Grenzfrequenz des RC-Filter.....	67
6.3.7 Übertragung des Tiefpassfilters.....	68
6.3.8 Z-Transformation.....	69
6.4.1 Abtasttheorem.....	70
6.4.2 sinc-Funktion.....	71
6.4.3 Zahl der Quantisierungskennlinien.....	75
6.4.4 Augenblickswert der Spannung.....	77
6.4.5 Frequenz des Integrierts.....	82
6.4.6 Impulsverhältnis.....	89
6.4.7 Umsetzungsgeschwindigkeit.....	91
6.4.8 Zeit erstes Taktes der Zweitaktintegration.....	93
6.4.9 Zeit zweites Taktes der Zweitaktintegration.....	94
6.4.10 Zahl des einlesungen Bits des Zählers.....	94
6.6.1 harmonisches Signal im Frequenzgebiet.....	105

Seznam tabulek

4.2.1 Wertetabelle für die Negation.....	17
4.2.2 Wertetabelle für die Konjunktion.....	17
4.2.3 Wertetabelle für die Disjunktion.....	18
4.2.4 Wertetabelle für die Subjunktion.....	18
4.2.5 Wertetabelle für die Antivalenz.....	18
6.4.1 BCD-Code.....	78
6.4.2 binärer Code.....	78
6.4.3 Wertetafel der Flip-Flop-Shaltung RS.....	81
6.4.4 Wertetafel der Flip-Flop-Shaltung JK.....	81
6.4.5 Wertetafel der Flip-Flop-Shaltung D.....	81
6.4.6 Gray-Code.....	82
6.4.7 höchstwertiges Bit.....	82
6.4.8 niedrigstwertiges Bit.....	87
6.4.9 Beispiele der numerischen Codes.....	87

1. Úvod

Tento projekt je velice důležitý pro německy mluvící studenty VUT, kteří se věnují se číslicovému zpracování signálu a nemají dostatečné znalosti češtiny, aby mohli studovat českou literaturu. Tento slovník jim pomůže při studování české literatury tak, že si mohou neznámý odborný termín najít v tomto slovníku a zjistit, co vlastně německy znamená.

Protože je číslicové zpracování signálu nejrychleji se rozvíjející obor v elektrotechnice, který se využívá nejen v této oblasti, věřím, že bude určitě nějakým způsobem využit.

Cílem práce bylo vytvořit německo-český slovník z oblasti matematiky, fyziky a číslicového zpracování signálu. Následně si vybrat vhodný software, kterým bude nejlépe tento slovník vytvořen a pomocí tohoto softwaru slovník realizovat v multimediální podobě.

Německo-český slovník z oblasti matematiky a fyziky obsahuje většinu základních pojmů na úrovni střední školy.

Slovník pojmů z oblasti číslicového zpracování signálu je rozdělen na části tak, jak prochází signál při jeho zpracování. Tyto části jsou: antialiasingový filtr, vzorkovač, kvantování, samotný analogově digitální převod, digitálně analogový převod a spektrální analýza, která není úplně součástí zpracování číslicového signálu, ale rozhodně je to obor velice důležitý jak pro práci s digitálním signálem, tak i pro jeho zpracování. Určitě nebylo možné v této práci vyjmenovat všechny pojmy z této oblasti. Snažil jsem se vybrat pojmy, které by měl určitě každý student znát před tím, než se začne této problematice věnovat hlouběji.

Rozvaha nad softwarovým řešením slovníku je v samostatné kapitole. Mým cílem bylo zkusit udělat program ve více vývojových prostředích a pak se subjektivně rozhodnout pro program, který mi nejvíce vyhovoval. Po rozboru jsem se rozhodl pro vývojové prostředí v C++ Builder.

Protože v zadání nebylo přesně uvedeno, jak má výsledek mé práce vypadat a nebyl ani uveden program, ve kterém mám tvořit, měl jsem ve své práci velký prostor pro vlastní tvořivost. Snažil jsem se program vytvořit co nejjednodušší, tak aby s ním uživatel mohl co nejlépe pracovat. Jeho vizuální formu jsem se snažil přizpůsobit různým slovníkům, které se vyskytují např. na internetu nebo jako různé programy tohoto charakteru.

2. Deutsches Alphabet

Buchstabe		Bedeutungen	
Schriftzeichen	Aussprache	in der Mathematik	in der Physik
a		oft erste Variable	die Beschleunigung
A		die Fläche 10 im Hexadezimalsystem	das Einheitensymbol für das Ampere
ä		-	-
Ä		-	-
b		oft zweite Variable	-
B		11 im Hexadezimalsystem	die magnetische Induktion
c		oft dritte Variable	die Lichtgeschwindigkeit
C		Menge der komplexen Zahlen das Formelzeichen für die Kombination	die elektrische Kapazität das Einheitensymbol für Coulomb
d		Differential- und Integralrechnung	die Dichte
D		13 im Hexadezimalsystem	die elektrische Flussdichte
e		die Eulersche Zahl	Elementarladung
E		14 im Hexadezimalsystem	die elektrische Feldstärke
f		allgemeine Funktion	die Frequenz
F		15 im Hexadezimalsystem	das Formelzeichen für die Kraft
g		geometrische Begriff Gon	die Fallbeschleunigung
G		-	Zeichen für die G-Parität
h		die Höhe in der Geometrie benutzt	Plancksche Wirkungsquantum
H		die Menge der hyperkomplexen Zahlen	die magnetische Feldstärke
i		die imaginäre Einheit	Formelzeichen für Augenblick der Stromstärke
I		Die Einheitsmatrix	Formelzeichen für Stromstärke
j		die imaginäre Einheit	die Gesamtdrehimpulsquantenzahl eines Elektrons in einem Atom
J		-	die Stromdichte
k		oft Konstante	die Boltzmann-Konstante
K		-	das Einheitensymbol für Kelvin
l		-	das Formelzeichen für Länge

L		-	das Formelzeichen für die Leuchtdichte
m		-	das Formelzeichen für Masse das Einheitsymbol für Meter
M		-	das Formelzeichen für das Drehmoment
n		oft eine Variable	der Brechungsindex
N		die Menge der Natürlichen Zahlen	das Einheitsymbol für das Newton
o		das Zeichen für den Umfang das Zeichen für die Mittelinie	-
O		-	-
ö		-	-
Ö		-	-
p		-	das Formelzeichen für den Druck
P		das Formelzeichen für die Wahrscheinlichkeit	das Formelzeichen für die Leistung
q		-	das Zeichen für die Ladung
Q		die Menge der rationalen Zahlen	das Formelzeichen für die Ladung
r		das Zeichen für den Ortsvektor das Zeichen für den Halbmesser	das Zeichen für den Reflexionsfaktor
R		die Menge der reellen Zahlen	das Formelzeichen für den elektrischen Widerstand
s		-	das Formelzeichen für den Weg
S		das Zeichen für die Mitte	die Einheit des elektrischen Leitwerts
ß		-	-
t		-	das Formelsymbol für die Zeit
T		-	das Formelsymbol für die Temperatur
u		oft das Zeichen für den Vektor	das Symbol für die atomare Masseneinheit Formalzeichen für Augenblick der elektrische Spannung
U		-	das Formelzeichen für die elektrische Spannung
ü		-	-
Ü		-	-
v		oft das Zeichen für den Vektor	das Formelzeichen für die Geschwindigkeit
V		Formelzeichen für das Volumen Benutzt	das Einheitsymbol für das Volt
w		das Zeichen für die Wahrscheinlichkeit	das Formelzeichen für die Energiedichte
W		-	das Formelzeichen für Energie das Einheitsymbol für das Watt

x		eine Gleichung für eine unbekannte Grösse	-
X		-	-
y		die zweite Unbekannte in Gleichungen	-
Y		-	das Formelzeichen für die Hyperladung
z		Formelzeichen für ganze Zahl benutzt	ein Maß für die Rotverschiebung
Z		die Menge der ganzen Zahlen	das Formelzeichen für den Wellenwiderstand

3. Griechisches Alphabet

Buchstabe		Bedeutungen	
Schriftzeichen	Aussprache	in der Mathematik	in der Physik
α	[alfa]	Oft ein Winkel	das Formelzeichen für die Dämpfungskonstante
A	[alfa]	-	-
β	[vita]	Oft ein Winkel	das Formelzeichen für die Phasenkonstante
B	[vita]	-	-
γ	[gama]	Oft ein Winkel	das Formelzeichen für die Fortpflanzungskonstante
Γ	[gama]		-
δ	[delta]	das Formelzeichen für die Variation	-
Δ	[delta]	das Formelzeichen für die Differenz	-
ε	[e psilon]		das Formelzeichen für die Dielektrizitätskonstante
E	[e psilon]	-	-
ζ	[dzE: ta]		
Z	[dzE: ta]	-	-
η	[E: ta]	-	das Formelzeichen für den Wirkungsgrad
H	[E: ta]	-	-
ϑ	[TE: ta]	-	das Formelzeichen für die Temperatur
Θ	[TE: ta]	das Formelzeichen für den Induktionfluss	-
ι	[iO: ta]	-	-
I	[iO: ta]	-	-
κ	[kappa]		das Formelzeichen für die elektrische Leitfähigkeit
K	[kappa]	-	-

λ	[lambda]	-	das Formelzeichen für die Wellenlänge
Λ	[lambda]	-	das Formelzeichen für den magnetische Leitwert
μ	[mi]	das Formelzeichen für den Mittelwert der zufällige Anordnung	das Formelzeichen für die Permeabilität
M	[mi]	-	-
ν	[ny:]	-	das Formelzeichen für die Viskosität
N	[ny:]	-	-
ξ	[ksi:]	-	-
Ξ	[ksi:]	-	-
o	[o mikron]	-	-
O	[o mikron]	-	-
π	[pi:]	die Ludolfsche Zahl	-
Π	[pi:]	das Produkt	
ρ	[rO:]	-	das Formelzeichen für die Dichte das Formelzeichen für den spezifischen elektrischen Widerstand
P	[rO:]	-	-
σ	[si: gma]	die Standardabweichung	
Σ	[si: gma]	die Summe	-
τ	[tau]	-	die Zeitkonstante
T	[tau]	-	-
υ	[y: psilon]	-	-
Y	[y: psilon]		
φ	[fi:]	das Formelzeichen für den Phasenwinkel	-
Φ	[fi:]	-	das Formelzeichen für den magnetische Induktionsfluß
χ	[Ci:]	-	-
X	[Ci:]	-	-
ψ	[psi:]	-	-
Ψ	[psi:]	-	-
ω	[O: mega]	-	das Formelzeichen für die Winkelgeschwindigkeit
Ω	[O: mega]	das Formelzeichen für die elementar Ereignis	das Einheitensymbol für das Ohm

[6]

4. Mathematik

4.1 Grundbegriffe

Addition

Die Addition ist eine der vier Grundrechenarten in der Arithmetik.

Die Zahlen, die addiert werden, heißen Summanden. Ihr Ergebnis heißt Summe.

$$\text{Summand} + \text{Summand} = \text{Summe}$$

Subtraktion

Die Subtraktion ist eine der vier Grundrechenarten in der Arithmetik.

Die Zahlen, die subtrahiert werden, heißen Differenzen. Ihr Ergebnis heißt Differenz.

$$\text{Minuend} - \text{Subtrahend} = \text{Differenz}$$

Multiplikation

Die Multiplikation ist eine der vier Grundrechenarten in der Arithmetik.

Das Ergebnis, gesprochen "a mal b", heißt Produkt.

$$\text{Multiplikant} \cdot \text{Multiplikator} = \text{Produkt}$$

Division

Die Division ist eine der vier Grundrechenarten der Arithmetik.

Der Term $a:b$ heißt Quotient.

$$\text{Dividend} : \text{Divisor} = \text{Quotient}$$

Der Divisor muss unbedingt ungleich 0 sein.

natürlichen Zahlen

Die natürlichen Zahlen sind die beim Zählen verwendeten Zahlen 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, usw. Oft wird auch zu den natürlichen Zahlen gerechnet.

Die Menge der natürlichen Zahlen, Formelzeichen: $\mathbb{N}$

ganzen Zahlen

Die ganzen Zahlen sind eine Erweiterung der natürlichen Zahlen.

Die ganzen Zahlen umfassen alle Zahlen ..., -2, -1, 0, 1, 2, ... und enthalten damit alle natürlichen Zahlen sowie deren negative Zahlen.

Die Menge der ganzen Zahlen, Formelzeichen: $\mathbb{Z}$

rationalen Zahlen

Eine rationale Zahl ist eine Zahl, die als Verhältnis zweier ganzer Zahlen dargestellt werden kann.

Die Menge der rationalen Zahlen, Formalzeichen:

 $\mathbb{Q}$

reellen Zahlen

Reelle Zahlen sind eine Erweiterung des Bereichs der rationalen Zahlen. Diese Erweiterung ist nötig, weil die rationalen Zahlen für manche Längen keine Maßzahl bereitstellen.

Die Menge der reellen Zahlen, Formalzeichen:

 $\mathbb{R}$

komplexen Zahlen

Die komplexen Zahlen erweitern den Zahlenbereich der reellen Zahlen derart, dass auch Wurzeln negativer Zahlen berechnet werden können.

Dies gelingt durch Einführung einer neuen Zahl i als Lösung der Gleichung $i^2 = -1$. Diese Zahl i wird auch als imaginäre Einheit bezeichnet.

Komplexe Zahlen werden meist in der Form $a + b \cdot i$

Die Menge der komplexen Zahlen, Formalzeichen:

 $\mathbb{C}$

4.2 Mathematische Logik

Negation

Die Formalzeichen p'

In der formalen Logik bezeichnet die Negation eine logische Operation. In der klassischen Logik wird bei der Ausführung der Negation zu einer Aussage p eine Aussage mit entgegengesetztem Wahrheitswert gebildet.

p	p'
0	1
1	0

Tabelle 4.2.1 Wertetabelle für die Negation

Konjunktion

$$p = a \wedge b$$

Eine Konjunktion bezeichnet in der Logik eine bestimmte Verknüpfung einer Aussage a mit einer Aussage b . In der klassischen Logik ist die Konjunktion zweier Aussagen genau dann wahr, wenn sowohl a als auch b wahr sind.

A	b	p
0	0	0

0	1	0
1	0	0
1	1	1

Tabelle 4.2.2 Wertetabelle für die Konjunktion

Disjunktion

$$p = a \vee b$$

Eine Disjunktion bezeichnet im Allgemeinen eine Trennung, Sonderung. In der Logik ist eine Disjunktion, auch Adjunktion genannt, das nicht ausschließende oder: Eine Aussage, die zwei Aussagen a und b mit Hilfe eines Disjunktors verknüpft, ist genau dann wahr, wenn mindestens eine der Aussagen a oder b wahr ist.

a	b	P
0	0	0
0	1	1
1	0	1
1	1	1

Tabelle 4.2.3 Wertetabelle für die Disjunktion

Subjunktion

$$p = p \Rightarrow q$$

Die Subjunktion bzw. der Subjunktore $\Rightarrow$ ist der wenn-dann-Junktore der dialogischen Aussagenlogik.

A	b	P
0	0	1
0	1	0
1	0	1
1	1	1

Tabelle 4.2.4 Wertetabelle für die Subjunktion

Antivalenz

$$p' = a \Leftrightarrow b$$

Antivalenz drückt in der Mathematik aus, dass die Verbindung von zwei Aussagen nur wahr wird, wenn sie unterschiedlich sind, das heißt, entweder die eine oder die andere wahr oder falsch ist, aber nicht beide gleichzeitig wahr oder falsch sind.

A	b	p'
0	0	0

0	1	1
1	0	1
1	1	0

Tabelle 4.2.5 Wertetabelle für die Antivalenz

4.3 Funktion

$$y = f(x)$$

Eine Funktion drückt die Abhängigkeit einer Größe von einer anderen aus. Traditionell werden Funktionen als Regel oder Vorschrift definiert, die eine Eingangsgröße (Argument, meist x) in eine Ausgangsgröße (Funktionswert, meist y) transformiert. Häufig werden auch die Begriffe Abbildung und Operation für Funktionen verwendet. Die Mathematik definiert Funktionen in den Begriffen der Mengenlehre.

konstante Funktion

In der Mathematik ist eine konstante Funktion eine Funktion, die für alle Argumente stets denselben, also konstanten Wert hat.

Linearfunktion

In der Mathematik ist eine konstante Funktion eine Funktion mit der Funktionsgleichung $y = ax + b$. Der Graph einer linearfunktion ist eine Linie.

quadratische funktion

Eine quadratische Funktion ist eine Funktion mit der Funktionsgleichung $y = ax^2 + bx + c$. Der Graph einer quadratischen Funktion ist eine Parabel.

Antilogarithmus

Es ist die Funktion in der Form $y = a^x$

Inversionsfunktion

Die Inversionfunktion zu der Funktion $y = f(x)$ über die Veränderliche $[c,d]$ ist die Funktion, die der Veränderliche $[d,c]$ angehört.

Logarithmusfunktion

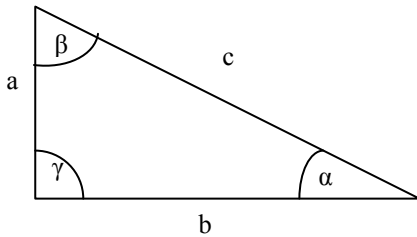
Es ist der Antilogarithmus $y = a^x$. Logarithmusfunktion ist die Inversfunktion zu dieser Funktion.

4.4 Trigonometrie des rechtwinkligen Dreiecks

Bodenmaß

Der ebene Winkel von 1 Radiant umschließt auf der Umfangslinie eines Kreises mit 1 Meter Radius einen Bogen der Länge 1 Meter. Der Vollwinkel umfasst 2π Radiant: 1 Vollwinkel = 2π rad.

goniometrische Funktion



$$\sin \alpha = \frac{a}{c} \quad (4.4.1)$$

wo: α ...der Winkelbetrag
a...die Gegenkathete
c... die Hypotenuse

$$\cos \alpha = \frac{b}{c} \quad (4.4.2)$$

wo: α ...der Winkelbetrag
b...die Ankathete
c... die Hypotenuse

$$\operatorname{tg} \alpha = \frac{a}{b} \quad (4.4.3)$$

wo: α ...der Winkelbetrag
a...die Gegenkathete
b...die Ankathete

$$\cot g \alpha = \frac{b}{a} \quad (4.4.4)$$

wo: α ...der Winkelbetrag
b...die Ankathete
a...die Gegenkathete

4.5 kongruente Darstellung im Ebene

Identität

Achsenspiegelung

Die Achsenspiegelung ist durch eine Gerade a gegeben. Sie ordnet jedem Punkt P einen Bildpunkt P' zu, der dadurch bestimmt ist, dass die Verbindungsstrecke $[PP']$ von der Achse a rechtwinklig halbiert wird.

Punktspiegelung

Es handelt sich um eine Abbildung, die durch einen Punkt Z gegeben ist. Die Spiegelung am Punkt Z ordnet jedem Punkt P der Zeichenebene oder jedem Punkt des Raumes einen Bildpunkt P' zu, der dadurch bestimmt ist, dass die Verbindungsstrecke $[PP']$ vom Punkt Z halbiert wird.

Drehung

Unter einer Drehung versteht man in der Geometrie eine eindeutige Abbildung einer Figur auf eine andere mit folgender Eigenschaft: Es gibt einen Punkt Z , das so genannte Drehzentrum, und einen Winkel α , den so genannten Drehwinkel, so dass für alle Punkte P der Ausgangsfigur und ihr Bild P' der Zielfigur gilt:

- 1) P und P' haben die gleiche Entfernung von Z : $\overline{PZ} = \overline{P'Z}$
- 2) Der Winkel $\angle PZP'$ ist gleich α

Parallelverschiebung

Die Parallelverschiebung oder Translation ist eine geometrische Abbildung, die jeden Punkt der Zeichenebene oder des Raumes in derselben Richtung um die selbe Strecke verschiebt. Sie kann durch einen Vektor, den so genannten Verschiebungsvektor, gekennzeichnet werden.

Sätze über die Kongruenz des Dreieckes

Lehrsatz SSS: Zwei Dreiecken sind kongruente, wenn sie in allen dreien Seiten zusammenfallen.

Lehrsatz SWS: Zwei Dreiecken sind kongruente, wenn sie in zweien Seiten und dem Winkel zwischen ihr zusammenfallen.

Lehrsatz WSW: Zwei Dreiecken sind kongruente, wenn sie in zweien Winkeln und der Seite zu ihnen anliegend zusammenfallen.

4.6 ähnliche Darstellung im Ebene

Zwei Gebilde sind ähnlich, wenn die ähnliche Darstellung im Ebene existiert.

Sätze über die Ähnlichkeit des Dreieckes

Lehrsatz SSS: Zwei Dreiecken sind ähnlich, wenn sie in allen Verhältnissen der Seiten zusammenfallen.

Lehrsatz WW: Zwei Dreiecken sind ähnlich, wenn sie in zwei Winkel zusammenfallen.

Lehrsatz SWS: Zwei Dreiecken sind ähnlich, wenn sie in einem Winkel und in dem Verhältnis der Seiten auf seinen Schenkeln zusammenfallen.

Euklidischer Lehrsatz über die Höhe

Der Fassungsraum des Viereckes, der über die Höhe konstruiert in dem rechtwinkligen Dreieck konstruiert wird, ist gleich dem Fassungsraum des Viereckes, der aus beiden Abschnitten der Hypotenuse konstruiert wird.

$$v^2 = c_a \cdot c_b \quad (4.6.1)$$

Euklidischer Lehrsatz über die Kathete

Der Fassungsraum des Viereckes, der über der Kathete in dem rechtwinkligen Dreieck konstruiert wird, ist gleich dem Fassungsraum des Viereckes, der aus dem Hypotenuse und dem Abschnitt der Hypotenuse konstruiert wird, der zu diese Kathete anliegend ist.

$$b^2 = c \cdot c_b \quad (4.6.2)$$

Pythagoreischer Lehrsatz

Der Fassungsraum des Viereckes, der über die Hypotenuse in dem rechtwinkligen Dreieck konstruiert wird, ist gleich der Summe der Fassungsräume der Vierecken über seinen Katheten.

4.7 Trigonometrie

Sinus Lehrsatz

Das Verhältnis der Länge der Seite und sin gegengesetzten Winkel ist in dem Dreieck unveränderlich.

$$\frac{a}{\sin \alpha} = \frac{b}{\sin \beta} = \frac{c}{\sin \gamma} \quad (4.7.1)$$

Cosinus Lehrsatz

ABC ist ein Dreieck mit den Seiten a, b, c und den Winkeln α , β , γ , dann es gilt:

$$\begin{aligned} a^2 &= b^2 + c^2 - 2bc \cdot \cos \alpha \\ b^2 &= a^2 + c^2 - 2ac \cdot \cos \beta \\ c^2 &= b^2 + a^2 - 2ba \cdot \cos \gamma \end{aligned} \quad (4.7.2)$$

4.8 komplexen Zahlen

Komplexen Zahlen sind Zahlen in der Form:

$$a + bi \quad (4.8.1)$$

wo: a...der Realteil
 b...der Imaginärteil
 i...die imaginäre Einheit

Die Abbildung der komplexen Zahl ist ein Punkt in der Ebene.

Gleichheit der komplexen Zahlen

Zwei komplexe Zahlen sind gleich, wenn Realteilen und Imaginärteilen gleich sind.

Addition der komplexen Zahlen

Zwei komplexe Zahlen addieren wir so, dass wir selbstständig Realteilen und Imaginärteilen addieren.

Subtraktion der komplexen Zahlen

Zwei komplexe Zahlen nehmen wir so ab, dass wir selbstständig Realteilen und Imaginärteilen abnehmen.

Multiplikation der komplexen Zahlen

Komplexen Zahlen multiplizieren wir als das Binom.

Division der komplexen Zahlen

Komplex konjugierte Zahl zu der Zahl $a = a_1 + a_2i$ ist die Zahl $b = a_1 - a_2i$.

Zwei komplexen Zahlen teilen wir so, dass wir die Zahl mit der komplex konjugierter Zahl weitem.

exponentiale Form der komplexen Zahlen

Absolutwert der komplexen Zahl $a = a_1 + a_2i$ verstehen wir die Zahl:

$$|a| = \sqrt{a_1^2 + a_2^2} \quad (4.8.2)$$

wo: |a|...der Absolutwert der komplexen Zahl
 a_1 ...der Realteil
 a_2 ...der Imaginärteil

Das Argument der komplexen Zahl $a = a_1 + a_2i$ verstehen wir die Zahl:

$$\operatorname{tg} \varphi = \frac{a_2}{a_1} \quad (4.8.3)$$

wo: φ ...das Argument der komplex Zahl

a_1 ...der Realteil
 a_2 ...der Imaginärteil

Die exponentiale Form der komplexen Zahl $a = a_1 + a_2i$ verstehen wir die Zahl:

$$a = |a| \cdot e^{i\varphi} \quad (4.8.4)$$

wo: a ...die komplexe Zahl
 $|a|$...der Absolutwert der komplexen Zahl
 e ...die eulersche Konstante
 i ...die imaginäre Einheit
 φ ...das Argument der komplexen Zahl

goniometrische Form der komplexen Zahlen

$$a = |a| \cdot (\cos \varphi + i \cdot \sin \varphi) \quad (4.8.5)$$

wo: a ...die komplexe Zahl
 $|a|$...der Absolutwert der komplexen Zahl
 i ...die imaginäre Einheit
 φ ...das Argument der komplexen Zahl

4.9 Logarithmus

Der Logarithmus y ist der Exponent, auf den wir die Grundzahl potenzieren müssen, damit wir logarithmische Zahl x bekommen.

$$y = \log_a x \quad (4.9.1)$$

wo: y ...der Logarithmus
 a ...die Grundzahl
 x ...die Logarithmische Zahl

Sätzen für die Rechnung mit dem Logarithmus

$$\log 1 = 0$$

$$\log(xy) = \log x + \log y$$

$$\log \frac{x}{y} = \log x - \log y$$

$$\log x^y = y \cdot \log x$$

4.10 Vektor

$$\vec{u} = (u_1, u_2, u_3)$$

Der Vektor ist die Menge von allen zustimmend orientierten Strecken, die gleiche Größe haben.

Gleichheit der Vektors

Zwei Vektors sind gleich, wenn entsprechende Koordinate gleich sind.

Größe der Vektors

$$|\vec{u}| = \sqrt{u_1^2 + u_2^2} \quad (4.10.1)$$

wo: u...Die Größe der Vektors
u₁...erste koordinate
u₂...zweite koordinate

Addition der Vectors

Zwei Vectors addieren wir so, dass wir jeweiligere Koordiante addieren.

inneres Produkt der Vestors

Das innere Produkt der Vectors ist eine reele Zahl in der Form:

$$\vec{u} \cdot \vec{v} = |\vec{u}| \cdot |\vec{v}| \cdot \cos \varphi \quad (4.10.2)$$

vektorielles Produkt der Vectors

Das vektorielle Produkt von zwei Vectors ist ein Vector in der Form:

$$\vec{u} \times \vec{v} = (u_2 v_3 - u_3 v_2; u_3 v_1 - u_1 v_3; u_1 v_2 - u_2 v_1) \quad (4.10.3)$$

4.11 analytische Geometrie

Parametergleichung der Linie in der Ebene

Der gerichteter Vector ist der Vector, der mit der Linie parallel ist.

$$\begin{aligned} x &= x_1 + tu_1 \\ y &= y_1 + tu_2 \end{aligned} \quad (4.11.1)$$

wo: x,y...der beliebig Punkt der Linie
x₁,y₁...der bekannten Punkt der Linie
u₁,u₂...der gerichteter Vector
t...der Parameter

Allgemeine Gleichung der Linie

Der Normalvektor ist der Vector, der zu der Linie senkrecht ist.

$$ax + by + c = 0 \quad (4.11.2)$$

wo: a,b,c...die Parametern
x,y...die Koordinate des bekannten Punktes

Parametergleichung der Ebene

$$\begin{aligned} x &= x_1 + tu_1 + sv_1 \\ y &= y_1 + tu_2 + sv_2 \\ z &= z_1 + tu_3 + sv_3 \end{aligned} \quad (4.11.3)$$

wo: x,y,z...der beliebige Punkt der Linie
x₁,y₁, z₁...der bekannten Punkt der Linie
u₁,u₂, u₃...der gerichtete Vector
t,s... die Parametern

Allgemeine Gleichung der Ebene

$$ax + by + cz + d = 0 \quad (4.11.4)$$

wo: a,b,c,d...die Parametern
x,y,z...die Koordinate des bekannten Punktes

4.11.1 Abweichung der Gebilde in den Raum

Abweichung von zwei Linien

Die Abweichung der Linien p und q ist eine Größe des Winkels, der beliebig ausgewählte Gipfel hat.

$$\cos \varphi = \frac{|\vec{u} \cdot \vec{v}|}{|\vec{u}| \cdot |\vec{v}|} \quad (4.11.5)$$

Abweichung zweier Ebenen

Die Abweichung der Eben p und q ist gleich der Abweichung der Schnittgerade p, q dieser Eben mit der Ebene τ, die zu beidem Ebenen senkrecht ist.

$$\cos \varphi = \frac{|\vec{n} \cdot \vec{m}|}{|\vec{n}| \cdot |\vec{m}|} \quad (4.11.6)$$

Abweichung der Linie und der Ebene

Die Abweichung der Ebene p und der Linie p ist gleich der Abweichung der Linie p und der Schnittgerade q der Ebene p mit der Ebene τ, die zu der Ebene p senkrecht ist und die Linie p beinhaltet.

$$\sin \varphi = \frac{\left| \begin{matrix} \vec{u} \cdot \vec{n} \end{matrix} \right|}{\left| \begin{matrix} \vec{u} \end{matrix} \right| \left| \begin{matrix} \vec{n} \end{matrix} \right|} \quad (4.11.7)$$

4.11.2 Kreislinie

Die Kreislinie ist die Menge von allen Punkten in der Ebene, die von dem Punkt S gleiche Distanz haben.

Zentrale Gleichung der Kreislinie

$$x^2 + y^2 = r^2 \quad (4.11.8)$$

wo: r...der Halbmesser
 x,y...die Koordinate des Punktes

Allgemeine Gleichung der Kreislinie

$$x^2 + y^2 + ax + by + c = 0 \quad (4.11.9)$$

wo: a,b,c...die Parametern
 x,y...die Koordinate des Punktes

4.11.3 Ellipse

Die Ellipse ist die Menge von allen Punkten in der Ebene, die von zwei Punkten F_1 und F_2 gleiche Summe der Distanzen haben, die größer als die Distanz von diesen Punkten ist.

Zentrale Gleichung der Ellipse

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1 \quad (4.11.10)$$

wo: a...die haupte Halbachse
 b...die unwesentliche Halbachse
 x,y...die Koordinate des Punktes

Allgemeine Gleichung der Ellipse

$$ax^2 + by^2 + cx + dy + e = 0 \quad (4.11.11)$$

wo: a,b,c,d,e...die Parametern
 x,y...die Koordinate des Punktes

4.11.4 Hyperbel

Die Hyperbel ist die Menge von allen Punkten in der Ebene, die, von zwei Punkten F_1 und F_2 gleiche Differenz der Distanz haben, die größer als die Distanz von diesen Punkten ist.

Zentrale Gleichung der Hyperbel

$$\frac{x^2}{a^2} - \frac{y^2}{b^2} = 1 \quad (4.11.12)$$

wo: a ...die haupte Halbachse
 b ...die unwesentliche Halbachse
 x, y ...die Koordinate des Punktes

Allgemeine Gleichung der Hyperbel

$$ax^2 - by^2 + cx + dy + e = 0 \quad (4.11.13)$$

wo: a, b, c, d, e ...die Parametern
 x, y ...die Koordinate des Punktes

4.11.5 Parabel

Die Parabel ist die Menge von allen Punkten in der Ebene, die gleiche Distanz von dem Punkt F und von der Linie d haben.

Scheitelform der Parabel

$$y^2 = 2px \quad (4.11.14)$$

wo: x, y ...die Koordinate des Punktes
 p ...der Parameter

Allgemeine Gleichung der Parabel

$$y^2 + ax + by + c = 0 \quad (4.11.15)$$

wo: a, b, c ...die Parametern
 x, y ...die Koordinate des Punktes

4.12 Folge

Die Zahlen $a_1, a_2, a_3 \dots a_n$ gestalten die Folge, wenn wir jeder Zahl n nach einer beliebigen Regel die Zahl a_n festlegen können.

Folge kann man festlegen:

Das Verzeichnis von allen Elementen

Die Formel für n-ten Element

Grafisch

Rekurrenzer – Wir führen ein die Regel an, nach der wir ein beliebig Element festlegen

wachsende und sinkende Folge

Die Folge ist wachsende, wenn $a_n < a_{n+1}$ gilt.

Die Folge ist sinkende, wenn $a_n > a_{n+1}$ gilt.

Die Folge ist nicht wachsende, wenn $a_n \leq a_{n+1}$ gilt.

Die Folge ist nicht sinkende, wenn $a_n \geq a_{n+1}$ gilt.

4.12.1 arithmetische Folge

Die Folge $\{a_n\}^\infty$ ist arithmetische, wenn die Differenz des beliebigen Elementes und des unmittelbaren vorangehenden Elementes gleich bleibende ist .

$$a_{n+1} = a_n + d \quad (4.12.1)$$

wo: a_{n+1} ...nächste Element der Folge
 a_n ...vorangehendes Element
 d ...die Differenz

Summe der arithmetischen Folge

$$s_n = \frac{n}{2} \cdot (a_1 + a_n) \quad (4.12.2)$$

wo: s_n ...die Summe der arithmetischen Folge
 a_1 ...erstes Element
 a_n ...lätztes Element

4.12.2 geometrische Folge

Die Folge $\{a_n\}^\infty$ ist geometrische, wenn die Quotient belibiges Elementes und unmittelbar vorangehendes Elementes gleich bleibende ist .

$$a_{n+1} = a_n \cdot q \quad (4.12.3)$$

wo: a_{n+1} ...nächstes Element der Folge
 a_n ...vorangehendes Element
 q ...die Differenz

Summe der geomertischen Folge

$$s_n = a_1 \cdot \frac{1 - q^n}{1 - q} \quad (4.12.4)$$

wo: s_n ...die Summe der arithmetischen Folge
 a_1 ...erstes Element
 a_n ...l tztes Element

4.12.3 Grenzwert der Folge

$\{a_n\}^\infty$ ist eine Folge, wir sagen, dass die Zahl a der Grenzwert dieser Folge ist, wenn zu jeder Zahl $\varepsilon > 0$ existiert n_0 so, dass f r alle n , wo $n \geq n_0$, $a_n - a < \varepsilon$ ist.

S tze  ber die Grenzwerten der Folge

- 1) Jede Folge hat nur ein Grenzwert.
- 2) Jede konvergente Folge ist begrenzt.
- 3) Jede begrenzte Folge ist konvergente.
- 4) Es gilt:

$$\lim_{n \rightarrow \infty} (a_n + b_n) = \lim_{n \rightarrow \infty} a_n + \lim_{n \rightarrow \infty} b_n$$

$$\lim_{n \rightarrow \infty} (a_n - b_n) = \lim_{n \rightarrow \infty} a_n - \lim_{n \rightarrow \infty} b_n$$

$$\lim_{n \rightarrow \infty} (a_n \cdot b_n) = \lim_{n \rightarrow \infty} a_n \cdot \lim_{n \rightarrow \infty} b_n$$

$$\lim_{n \rightarrow \infty} (c \cdot b_n) = c \cdot \lim_{n \rightarrow \infty} b_n$$

- 5) Die Folge $\left\{\frac{1}{n}\right\}_{n=1}^\infty$ hat den Grenzwert 0.

unendliche Reihe

Es ist die Folge $\{a_n\}_{n=1}^\infty$, dann wir den Begriff $a_1, a_2, a_3 \dots a_n$ die unendliche Reihe nennen.

4.13 Differenzrechnung und Integralrechnung

Grenzwert der Funktion

Es ist die Funktion $f(x)$, die Zahl x r ckt grenzenlos zu der Zahl c . Wenn die Wert der Funktion $f(x)$ grenzenlos zu dem Wert der Zahl c r ckt, wir sagen, dass die Funktion $f(x)$ in dem Punkt c einen Grenzwert hat.

S tze  ber die Grenzwerten der Funktion

- 1) Die Funktion hat in dem Punkt nur ein Grenzwert.
- 2) Der Grenzwert der Konstante ist die Konstante.
- 3) Wenn in dem Punkt A kontinuierliche Funktion ist, der Grenzwert ist
 $\lim_{x \rightarrow a} f(x) = f(a)$

- 4) Es gilt:

$$\begin{aligned}\lim_{n \rightarrow a} [f(x) + g(x)] &= \lim_{n \rightarrow a} f(x) + \lim_{n \rightarrow a} g(x) \\ \lim_{n \rightarrow a} [f(x) - g(x)] &= \lim_{n \rightarrow a} f(x) - \lim_{n \rightarrow a} g(x) \\ \lim_{n \rightarrow a} [f(x) \cdot g(x)] &= \lim_{n \rightarrow a} f(x) \cdot \lim_{n \rightarrow a} g(x) \\ \lim_{n \rightarrow a} c \cdot f(x) &= c \cdot \lim_{n \rightarrow a} f(x)\end{aligned}$$

Derivation der Funktion

Die Derivation ist die Richtlinie der Berührungslinie zu dem Graph in dem Punkte

$$y' = \lim_{\Delta x \rightarrow 0} \frac{f(x_0 + \Delta x) - f(x_0)}{\Delta x} \quad (4.13.1)$$

wo:
 y' ... die Derivatin der Funktion
 x_0 ... der Anfangspunkt
 Δx ... die Verschiebung

Rechnung der Derivation

- 1) Die Derivation der Konstante ist 0.
- 2) Die Derivation der Summe ist die Summe der Derivationen
- 3) Die Derivation der Diferenz ist die Diferenz der Derivationen
- 4) Für die Derivation des Produktes gitl es:

$$[f(x) \cdot g(x)]' = f(x)' \cdot g(x) + f(x) \cdot g(x)'$$

- 5) Für Derivation des Anteil gilt es:

$$\left[\frac{f(x)}{g(x)} \right]' = \frac{f(x)' \cdot g(x) - f(x) \cdot g(x)'}{g(x)^2}$$

Derivation der zusammengesetzten Funktion

Die Derivation der zusammengesetzten Funktion ist der Produkt der einfachen Funktion, auf die die zusammengesetzte Funktion gliedern kann.

Zweite und hohe Derivation

Hohe Derivation rechnen wir gleich als erste derivation. Die Anzahl der Derivation bestimmt der Rang der Derivation.

Differential der Funktion

Der Begriff dy heißen wir das Differential der Funktion und wir definieren ihn als Produkt der Derivation der Funktion über das Differential des Argumentes dx .

uneigentliches Integral

Integration ist verkehrte Funktion zu der Derivation. Zu dem Diferenzial der Funktion suchen wir diese Funktion durch Integrieren.

Das Ergebnis der Integration ist primitive Funktion in der Form:

$$\int f(x) dx = f(x) + c \quad (4.13.2)$$

wo: $\int f(x) dx$...das Integral der Funktion
 $f(x)$...die Funktion
 c ...die Konstante

Sätze für die Rechnung mit den Integralen

1) Es gilt:

$$\int k \cdot f(x) dx = k \cdot \int f(x) dx$$

2) Das Integral der Summe ist gleich wie die Summe der Integralen

3) Es existiert keine Formel für die Integration des Produktes und des Anteiles.

bestimmtes Integral

Das bestimmtes Integral Rechnen wir so, dass wir in das uneigentliches Integral zuerst obere Grenze und dann untere Grenze substituieren. Beide Worthen in dieser Reihung werden wir abrechnen.

$$\int_a^b f(x) dx = [f(x)]_a^b = f(b) - f(a)$$

Eigenschaften des bestimmtes Integral

$$1) \int_a^a f(x) = 0$$

$$2) \int_a^b f(x) = - \int_b^a f(x)$$

$$3) \int_a^b f(x) + \int_b^c f(x) = \int_a^c f(x)$$

4.14 Matrizen

Zahlen m, n sind ganzen Zahlen und es gibt die Zahl a_{ik} ($i = 1, 2, 3 \dots m, k = 1, 2, 3 \dots n$), dann das Schema, das aus den Zahlen a_{ik} , die in die m Zeilen und n Spalten zusammengeleift werden, nennen wir die Matrize des Typs (m,n) .

Speziell Matrize

Die Nullmatrix ist die Matrize, die alle Elemente 0 hat.

Die quadratische Matrize ist die Matrize des Typs (n,n) .

Die Einheitsmatrize ist eine quadratische Matrize, die diagonale Elemente 1 hat und andere Elemente 0 sind.

Die Dreiecksmatrize ist die Matrize, die die Elemente der Hauptdiagonale verschiedene von 0 hat, und unter der Hauptdiagonale 0 sind.

Die transponierte Matrize ist die Matrize, die von dem Wechsel der Zeilen für die Spalten entsteht.

4.14.1 Operation mit der Matrizen

Gleichheit der Matrizen

Zwei Matrizen sind gleich, wenn entsprechende Elemente gleich sind.

Addition der Matrizen

Die Sume der Matrize A und der Matrize B ist die Matrize C, für ihre Elemente gilt es:

$$c_{ik} = a_{ik} + b_{ik}$$

Produkt der Zahl und der Matrizen

Der Produkt der Zahl c und der Matrize A ist die Matrize B, für ihre Elemente gilt es:

$$b_{ik} = c \cdot a_{ik}$$

Produkt der Matrizen

Der Produkt der Matrize $A = [a_{ik}]_m^n$ und der Matrize $B = [b_{ik}]_n^p$ in dieser Reihung nennen wir die Matrize $C = [c_{ik}]_p^m$, die die Elemente c_{ik} hat, die inneres Produkt der i-ten Zeile und der k-ten Spalten ist.

4.15 Determinante

Die Determinant n Rang der quadratischen Matrize A_n nennen wir die Zahl a und die Scheme

$$a = \det A = \begin{vmatrix} a_{11} & a_{12} \dots & a_{1n} \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ a_{n1} & a_{n2} \dots & a_{nn} \end{vmatrix}$$

Unterdeterminante

Die Unterdeterminante (n-k) Rang entsteht aus der Determinant n Rang aus dem Übergang k Zeilen und k Spalten.

Laplacetransformierte

A ist die Determinante n-tes Rang, dann es gilt:

$$A = \sum_{i=1}^n a_{i1}D_{i1} + a_{i2}D_{i2} + \dots + a_{in}D_{in} \quad (4.15.1)$$

Cramersch Regel

Wenn die Determinante des Gleichungssystems $D \neq 0$, dann hat dieses Gebilde eine Lösung

$$x_i = \frac{D_i}{D} \quad (4.15.2)$$

wo: x_i ...die Lösung des Gleichungssystems

D_i ...die Determinante, die nach der Substitution i Spalte durch die Zahlen auf der linken Seite des

Gleichungssystem entsteht

D ...die Determinante des Gleichungssystems

4.16 Kombinatorik

Variation

Die Variation der k Rang aus den n Elementen ist jede veranstaltete Menge, aus diesen Elementen zusammengesetzt wird so, dass jedes Element in ihr nur einmal enthalten ist.

$$V_k = n \cdot (n-1) \cdot (n-2) \cdot \dots [n - (k-1)] \quad (4.16.1)$$

wo: V_k ...die Variation

n ... die Zahl von den Elementen

k ...die Rang

Variation mit der Rekursion

Die Variation der k Rang mit der Rekursion aus n Elementen ist jede veranstaltete Menge aus diesen Elementen aneinanderfügen wird so, dass jedes Element in ihr höchstens k-mal enthaltene ist.

$$V_{k(n)} = n^k \quad (4.16.2)$$

wo: V_k ...die Variation mit der Rekursion

n ... die Zahl von den Elementen

k ...die Rang

Permutation

Die Permutation aus den n Elementen ist jede n-elemente Variation aus diesen Elementen.

$$P_{(n)} = n! \quad (4.16.3)$$

wo: P ...die Permutation

n ... die Zahl den Elementen

Kombination

Die Kombination ist die Menge von allen Variationen, die gleiche Elemente beinhalten.

$$C_{k(n)} = \frac{n!}{k!(n-k)!} \quad (4.16.4)$$

wo: C_k ...die Variation mit der Rekursion
n... die Zahl von den Elementen
k...die Rang

binomischer Lehrsatz

Für beliebige Zahlen a, b, n gilt es:

$$(a+b)^n = \binom{n}{0} \cdot a^n + \binom{n}{1} \cdot a^{n-1} \cdot b + \binom{n}{2} \cdot a^{n-2} \cdot b^2 + \binom{n}{3} \cdot a^{n-3} \cdot b^3 + \dots + \binom{n}{n} \cdot a^{n-n} \cdot b^n \quad (4.16.5)$$

4.17 Wahrscheinlichkeit

Die elementare Ereignisse sind einzeln Ergebnisse des Experimentes der Mengen von elementaren Ereignisse. Wir Ziehen ihr Ω .

Wahrscheinlichkeit des zufälligen Ereignisses

Es gilt:

- 1) Jedem zufällig Ereignis wird nichtnegative Wahrscheinlichkeit $P_{(A)}$ zugeordnet, wo $0 \leq P_{(A)} \leq 1$
- 2) Die Wahrscheinlichkeit der Vereinigung der unverträglich Ereignis ist die Summe c dieser Wahrscheinlichkeiten.
- 3) Die Wahrscheinlichkeit des sichere Ereignisses ist 1. Die Wahrscheinlichkeit des unmöglichen Ereignisses ist 0.

$$P_{(A)} = \frac{m}{n} \quad (4.17.1)$$

wo: $P_{(A)}$...die Wahrscheinlichkeit
m...die Zahl der positiven Ereignisse
n...die Zahl von allen Ereignissen

Wahrscheinlichkeit der Vereinigung der Ereignis

Wenn die Ereignisse A und B gegenseitig absondern, es gilt:

$$A \cup B = P_{(A)} + P_{(B)}$$

unabhängige Ereignisse

Es sind Ereignisse, wenn das Anfangen keinen Einfluss auf das Anfangen von dem zweiten hat Ereignis.

Die Ereignisse sind unabhängig, wenn es gilt: $P_{(A \cup B)} = P_{(A)} \cdot P_{(B)}$

$$s = v_0 \cdot t + \frac{1}{2} a \cdot t^2 \quad (5.1.3)$$

wo: s... die Bahn der kontinuierlich beschleunigten Bewegung
 v_0 ...die Bezugsgeschwindigkeit
t...die Zeit
a... der Beschleunigungen

Freifall

Newton bewies, dass alle Körper ohne den Luftwiderstand mit gleichem Beschleunigungen fallen.

Er hat die Gravitationskonstante g eingeführt. $g = 9,81 \text{ m.s}^{-2}$

$$v = g \cdot t \quad (5.1.4)$$

wo: v...die Schlussgeschwindigkeit
g...die Gravitationskonstante
t... die Zeit

$$s = \frac{1}{2} g \cdot t^2 \quad (5.1.5)$$

wo: s... die Bahn
g...die Gravitationskonstante
t... die Zeit

Rundbewegung

Die Winkelgeschwindigkeit

Zeichen: ω Einheit: Radiant je Sekunde Einheitensymbol: s^{-1}

$$\omega = \frac{2\pi}{T} \quad (5.1.6)$$

wo: ω ...die Winkelgeschwindigkeit
T...die Periode

In der Zeit t schreibt die Verbindungslinie der Mitte mit einem bestimmtem Punkt den Winkel ϕ um. Wenn wir diesen Winkel durch die Zeit t dividieren, bekommen wir die Winkelgeschwindigkeit.

Die Winkelgeschwindigkeit ist für ganzen Körper gleich.

5.2 Dynamik

Dynamik erklärt die Bewegung als die Folge der einwirkenden Kraft. Die Kraft ist die Wechselwirkung gegen die Körper: a) aufrechter Verkehr

b) unter der Wirkung des Kraftfeldes

Erstes Trägheitsgesetz

Der Körper verbleibt in der Ruhe oder in kontinuierlicher Linearbewegung, falls die äussere Kräfte diesen Stand nicht ändern.

Zweites Trägheitsgesetz

Der Beschleunigungen des Körpers ist direkt dem Kraft proportional und indirekt dem Gewicht proportional.

$$F = m \cdot a \quad (5.2.1)$$

wo: F...die Kraft
m...das Gewicht
a... der Beschleunigungen

Drittes Trägheitsgesetz

Das Gesetz der Wirkung und Gegenwirkung
Zwei Körper wirken auf sich immer mit den Kräften der gleichen Größen, aber der verkehrten Richtung.

Beweglichkeit des Körper und der Erhaltungssatz der Beweglichkeit

Die Beweglichkeit ist das Produkt des Gewichtes m und der Augenblicksgeschwindigkeit v.

Zeichen: p Einheit: Kilogramm Meter je Sekunde Einheitensymbol: kg.m.s⁻¹

$$p = m \cdot v \quad (5.2.2)$$

wo: p...die Beweglichkeit
m...das Gewicht
v...die Geschwindigkeit

Für das System des Körpers, das keine äusseren Kräfte beeinflussen, bleibt die ganze Beweglichkeit unverändert.

Arbeit

Die Arbeit ist in der physikalischen Bedeutung die Kraftwirkung auf den Körper, bei der zum Vorschub dieses Körper zugeht.

Zeichen: W Einheit: Joule Einheitensymbol: J

$$W = F \cdot s \quad (5.2.3)$$

wo: W... die Arbeit
F... die Kraft
s... die Bahn

Leistung

Die Leistung ist die geleistete Arbeit in einer Zeiteinheit.

Zeichen: P

Einheit: Watt

Einheitensymbol: W

$$P = \frac{W}{t} \quad (5.2.4)$$

wo: P...die Leistung
W... die Arbeit
t...die Zeit

Wirksamkeit

$$\eta = \frac{P_2}{P_1} \quad (5.2.5)$$

wo: η ...die Wirksamkeit
P₂... die Leistung
P₁...die Leistungsbedarf

5.2.1 Energie

Die Energie ist das Vermögen eine Arbeit zu machen. Die Energie hat die gleiche Einheit als die Arbeit.

Zeichen: Q

Einheit: Joule

Einheitensymbol: J

potenzielle Energie

$$E_p = m \cdot g \cdot h \quad (5.2.6)$$

wo: E_p...die Potenzielle Energie
m...das Gewicht
g...die Gravitationskonstante
h...die Höhenlage

kinetische Energie

$$E_k = \frac{1}{2} \cdot m \cdot v^2 \quad (5.2.7)$$

wo: E_k...die Kinetische Energie
m...das Gewicht
v...die Beweglichkeit

5.3 Mechanik der starken Körper

Es geht über die Körper, bei denen man die Deformation vernachlässigen kann.

Kraftmoment

Die drehbare Wirkung der Kraft hebt sich auf, wenn die Summe der Moments gleich Null ist.

Zeichen: M

Einheit: Newton Meter

Einheitensymbol: N.m

$$M = F \cdot a \quad (5.3.1)$$

wo: M...das Kraftmoment
F...die Kraft
a...die Länge des Arm

Druck

Der Druck ist direkt dem Kraft proportional und indirekt dem Fassungsvermögen proportional.

Zeichen: p

Einheit: Pascal

Einheitensymbol: Pa

$$p = \frac{F}{S} \quad (5.3.2)$$

wo: p...der Druck
F...die Kraft
S...Fassungsvermögen

5.4 Mechanik der Flüssigkeit

Eine Ideale Flüssigkeit fließt ohne die Auflehnung und sie ist absolut inkompressibel.
Ein Ideales Gas fließt ohne die Auflehnung und es ist komprimierbar bis auf den Nullrauminhalt.

Pascalches Gesetz

Einen Druck, den eine äussere Kraft macht ist in allen Stellen der Flüssigkeit gleich.

hydrostatischer Druck

Er ist wirksam durch eigenen Druck der Flüssigkeit
Der hydrostatische Druck wächst mit der Tiefe.

$$p_H = h \cdot \rho \cdot g \quad (5.4.1)$$

wo: p_H ...der Hydrostatische Druck
h...der Tiefe
 ρ ...die Dichte
g...die Gravitationskonstante

5.5 Elektrizität

elektrisches Feld

Das elektrische Feld gibt die Kraft an, durch die Feld auf eine Ladung bereitet.

Zeichen: E Einheit: Volt je Metr Einheitensymbol: $V \cdot m^{-1}$

$$E = \frac{F}{Q} \quad (5.5.1)$$

wo: E...das elektrische Feld
F...die Kraft
Q...die Ladung

elektrische Spannung

Est ist die Arbeit, die wir machen, wenn wir die Ladung aus dem Punkt A in den Punkt B übertragen.

Zeichen: U Einheit: Volt Einheitensymbol: V

$$U = \frac{W}{Q} \quad (5.5.2)$$

wo: U...die elektrische Spannung
W...die Arbeit
Q...die Ladung

Kapazität

Die Kapazität ist direkt der Ladung proportional und indirekt der elektrischen Spannung proportional.

Zeichen: C Einheit: Farad Einheitensymbol: F

$$C = \frac{Q}{U} \quad (5.5.3)$$

wo: C...die Kapazität
Q...die Ladung
U...die elektrische Spannung

Aus den Proportionen des Kondensators

$$C = \varepsilon_0 \cdot \varepsilon_r \cdot \frac{S}{l} \quad (5.5.4)$$

wo: C...die Kapazität
 ε_0 ...die Permittivität des Vakuums
 ε_r ...die Permittivität des Materiales
S... der Fassungsraum der Schaftelektrode
l...die Elektrodenentfernung

Coulombsches Gesetz

Die Kraft, mit der sich zwei Körper anziehen, fällt mit der zweiten Potenz der Distanz.

$$F = k \cdot \frac{Q_1 \cdot Q_2}{r^2} \quad (5.5.6)$$

wo: F...die Kraft
k...der Verhältnissfaktor
 Q_1, Q_2 ...die Ladung
r...die Distanz

elektrischer Strom

Der elektrishe Strom ist die Ladung, die des Draht in einer Zeiteinheit durchläuft.

Zeichen: I Einheit: Ampere Einheitensymbol: A

$$I = \frac{Q}{t} \quad (5.5.7)$$

wo: I...der elektrischer Strom
Q...die Ladung
t...die Zeit

Ohm Gesetz

Der elektrische Strom ist direkt der elektrische Spannung proportional und indirekt den elektrischen Widerstand proportional.

Der elektrischer Widerstand: Zeichen: R Einheit: Ohm Einheitensymbol: Ω

$$I = \frac{U}{R} \quad (5.5.8)$$

wo: I...der elektrischer Strom

U...die elektrische Spannung
R...der elektrischer Widerstand

Aus den Proportionen des Drahtes

$$R = \rho \cdot \frac{l}{S} \quad (5.5.9)$$

wo: R...der elektrischer Widerstand
ρ...der spezifischer Widerstand
l...die Länge
S...der Fassungsraum

Arbeit und die Leistung des elektrischen Stroms

Zeichen: W Einheit: Joule Einheitensymbol: J

$$W = U \cdot I \cdot t \quad (5.5.10)$$

wo: W...die Arbeit
I...der elektrischer Strom
U...die elektrische Spannung
t...die Zeit

Zeichen: P Einheit: Watt Einheitensymbol: W

$$P = U \cdot I \quad (5.5.11)$$

wo: P...die Leistung
I...der elektrischer Strom
U...die elektrische Spannung

5.6 Magnetismus

Form des magnetisches Feldes

Das Magnetische Felde stellen die Kraftlinien dar. Die Feldliniendichte dieser Linien prägt die Feldkraft.

Diamagnetische Materials

Sie haben unmagnetische Atome.

Paramagnetische Materials

Sie haben magnetische Atome, aber ihr magnetisches Feld kann man nicht in eine Richtung ordnen.

Feromagnetische Materials

Sie haben magnetische Atome, ihre magnetische Feld kann man in eine Richtung ordnen.

Intensität des Magnetischen Feldes

Die Intensität des Magnetischen Feldes ist die Magnetisierungsspannung auf eine Länge der Induktionslinie bezogen.

Zeichen: H

Einheit: Ampere je Meter

Einheitensymbol: A.m⁻¹

$$H = \frac{N \cdot I}{l} \quad (5.6.1)$$

wo: H... die Intensität des Magnetisches Feldes
I... der elektrischer Strom
l... die Länge der Induktionslinie
N... die Windungszahl

Magnetische Flussdichte

Faradaysches Gesetz

In dem Draht induziert man eine elektrische Spannung, wenn es sich in seiner Umgebung das magnetische Feld ändert.

Es ist die Zahl der Induktionslinien, auf eine Einheit der Fläche.

Zeichen: B

Einheit: Tesla

Einheitensymbol: T

$$B = \frac{\Phi}{S} \quad (5.6.2)$$

wo: B... die magnetische Flussdichte
Φ... der Induktionsfluss
S... der Fassungsraum

Selbstinduktivität der Spule

Es ist die Fähigkeit, die Spannung durch die Änderung des Eigenstrom zu induzieren.

Zeichen: L

Einheit: Henry

Einheitensymbol: H

$$u = L \cdot \frac{\Delta I}{t} \quad (5.6.3)$$

wo: u... die elektrische Spannung
L... die Selbstinduktivität
ΔI... die Änderung des Eigenstrom
t... die Zeit

Aus der Proportion der Spule

$$L = \mu_0 \cdot \mu_r \cdot \frac{S}{l} \cdot N^2 \quad (5.6.4)$$

wo: L... die Selbstinduktivität
 μ_0 ...die Permeabilität des Vakuums
 μ_r ...die Permeabilität des Materiales
S...der Kernquerschnitt
l... die Länge
N...die Windungszahl

5.7 Optik

Das Licht = die elektromagnetische Wellenbewegung.

Wellenlänge

Zeichen: λ Einheit: Meter Einheitsymbol: m

$$\lambda = \frac{v}{f} \quad (5.7.1)$$

wo: λ ... die Wellenlänge
v...die Lichtgeschwindigkeit
f...die Frequenz

Widerschein

- 1) der Reflexionswinkel = dem Inzidenzwinkel ($\alpha = \alpha'$)
- 2) Der Widerstrahl liegt in der Einfallsebene

5.7.1 Brechungsgesetz

Refraktionskoeffizient

Zeichen: n Einheit: - Einheitsymbol: -

$$n = \frac{c}{v} \quad (5.7.2)$$

wo: n... der Refraktionskoeffizient
c...die Lichtgeschwindigkeit in das Vakuum
v... die Lichtgeschwindigkeit in das Material

Brechungsgesetz

$$\sin \alpha \cdot n_1 = \sin \beta \cdot n_2 \quad (5.7.3)$$

wo: α ... der erste Winkel
 n_1 ... der erste Refraktionskoeffizient
 β ... der zweite Winkel
 n_2 ... der zweite Refraktionskoeffizient

Lichtbrechung auf der Grenzscheide

Es ist die Folge der Änderung der Geschwindigkeit.

- 1) Die Geschwindigkeit erhöht sich $\Rightarrow$ die Bruchung von der Senkrechte ($n_1 > n_2$)
- 2) Die Geschwindigkeit erhöht sich $\Rightarrow$ die Bruchung zu der Senkrechte ($n_2 > n_1$)

5.7.2 Wellenwesen der Leuchte

Dispersion des Lichtes

Auf den hochbrechenden Kantensäulen brecht sich jede Farbe des Lichtes anders.
Der Brechwert wächst mit der Frequenz des Lichtes.

Interferenz der Licht auf einer dünnen Schicht

Wenn auf eine dünne Schicht das Licht einfällt, sehen wir lichte und dunkle Plätze.

Wir haben zwei Möglichkeiten: 1) die Verschiebung über die ganze Wellenlänge – maximale Verstärkung
2) die Verschiebung über die halbe Wellenlänge – maximale Schwächung

Interferenz auf dem Beugungsgitter

Die Durchgang des Lichtes durch das Beugungsgitter verursacht seine Brechunge.

Lichtpolarisation

Die Polarisation beweist, dass das Licht die Querwellen ist. Natürliches Licht schwingt in allen Richtungen gleich. Manche Materiale können das Licht polarisieren.

Teilweise Polarisation – Die Schwindungen in eine Richtung vorwiegen.
Ganze Polarisation – Die Schwindungen sind nur in eine Richtung.

Abbildung in dem Planspiegel

Das Bild hat eine gleiche Größe, ist nicht verkehrt, aber es ist seitenverkehrt umgekehrt.

5.7.3 Abbildung in dem Kugelspiegel

Konkavspiegel

Grunde Strahlen

- 1) Der Strahl, der mit der Mittellinie parallel läuft spiegelt sich von dem Brennpunkt.
- 2) Der Strahl, der den Brennpunkt durchgeht, spiegelt sich parallellaufend mit der Mittellinie.

Das Objekt ist zwischen dem Linsenglas und dem Brennpunkt: Das Bild ist vergrößert, scheinbar und nicht verkehrt.

Das Objekt ist hinter dem Brennpunkt: Das Bild ist wirklich und verkehrt.

Wölbspiegel

Grunde Strahlen

- 1) Der Strahl, der in den Brennpunkt richtet, spiegelt sich als ob aus dem Brennpunkt.
- 2) Der Strahl, der in den Brennpunkt Richter, spiegelt sich parallellaufend mit der Mittellinie.

Das Bild ist immer verringert, scheinbar und nicht verkehrt.

Abbildungsgleichung

$$\frac{1}{a} + \frac{1}{a'} = \frac{1}{f} \quad (5.7.4)$$

wo: a... der Abstand des Gegenstandes
a'... der Abstand des Bildes
f...die Brennweite

5.8 Thermodynamik

Nullter Hauptsatz

Wenn ein System A sich mit einem System B sowie B sich mit einem System C im thermischen Gleichgewicht befindet, so befindet sich auch A mit C im thermischen Gleichgewicht.

Erster Hauptsatz

Jedes System besitzt eine innere Energie. Diese kann sich nur durch den Transport von Energie in Form von Arbeit und Wärme über die Grenze des Systems ändern

$$U = Q + W \quad (5.8.1)$$

wo: U...die innere Energie
Q...die Wärme
W...die Arbeit

Zweiter Hauptsatz

Der zweite Hauptsatz der Thermodynamik besagt, dass es eine extensive Zustandsgröße Entropie S gibt, die in einem abgeschlossenen System niemals abnimmt.

$$S = \frac{Q}{T} \quad (5.8.2)$$

wo: S ...die Entropie
 Q ...die Wärme
 T ...die Temperatur

Dritter Hauptsatz

Bei der Annäherung der Temperatur an den absoluten Nullpunkt ($T = 0$) wird die Entropie S unabhängig von thermodynamischen Parametern. Damit geht S gegen einen festen Grenzwert S_0 .

isothermischer Vorgang

Es ist der Vorgang, bei dem sich der Druck und das Volumen des Gases ändern und die Temperatur gleich bleibt.

isobarischer Vorgang

Es ist der Vorgang, bei dem sich das Volumen und die Temperatur des Gases ändern und der Druck gleich bleibt.

isochorer Vorgang

Es ist der Vorgang, bei dem sich der Druck und die Temperatur ändern und das Volumen gleich bleibt.

adiabatischer Vorgang

Es ist der Vorgang, bei dem sich der Druck, die Temperatur und das Volumen des Gases ändern.

Wärme

Die Wärme ist ein Teil der inneren Energie, die der Körper bei dem thermischen Austausch des zweiten Körpers nimmt an oder überreicht

Zeichen: Q Einheit: Joule Einheitensymbol: J

$$Q = m \cdot c \cdot \Delta T \quad (5.8.3)$$

wo: Q ...die Wärme
 m ...die Masse
 c ... Wärmekapazität
 T ...die Temperatur

Temperatur

Zeichen: T Einheit: Kelvin, Celsiusgrad Einheitensymbol: K, °C

Hohe Temperaturen bezeichnet man als heiß, niedrige als kalt.

Volumen

Zeichen: V Einheit: Kubikmeter Einheitsymbol: m³

In der Physik bezeichnet man mit dem Volumen die Ausdehnung eines Stoffes.

Schmelzwärme

Zeichen: L_t Einheit: Joule Einheitsymbol: J

$$L_t = l_t \cdot m \quad (5.8.4)$$

wo: L_t...die Schmelzwärme
 m...die Masse
 l_t... Die spezifische Schmelzwärme

Schmelzwärme bezeichnet die Energie, die benötigt wird, um eine Stoffprobe von dem festen in den flüssigen Aggregatzustand zu überführen.

Kristallisationswärme

Zeichen: L_t Einheit: Joule Einheitsymbol: J
Kristallisationswärme wird freigesetzt, wenn ein Stoff seinen Aggregatzustand von flüssig nach fest ändert.

spezifische Schmelzwärme

Zeichen: l_t Einheit: Joule pro Kilogramm Einheitsymbol: J.kg⁻¹

Die spezifische Schmelzwärme bezeichnet die Menge Energie, die zum Schmelzen eines Stoffes benötigt wird, bezogen entweder auf die Stoffmenge oder auf die Masse.

Schmelzen

Als Schmelzen bezeichnet man das Übergehen eines Stoffes vom festen in den flüssigen Aggregatzustand.

Verdampfen

Als Verdampfen oder auch Sieden bezeichnet man den Übergang eines Reinstoffes vom flüssigen in den gasförmigen Aggregatzustand am Siedepunkt und eines Stoffgemisches im Siedebereich.

6 DSP

6.1 Deutsch-Tschechisch Wörterbuch für DSP

6.1.1 Antialiasing

der aktive Filter	aktivní filtr
der Amplitudengang	amplitudová charakteristika
der analoge Filter	analogový filtr
der Analogsignal	analogový signál
der Antialiasing	Antialiasing
die Bandbreite	šířka pásma
der Bandpassfilter	pásmová propust
der Bandstopfilter	pásmová zádrž
der Butterworthfilter	Butterworthfiltr
die Differentialgleichung	diferenciální rovnice
der digitale Filter	digitální filtr
der Dirac-Impuls	diracův impuls
die diskrete Fourier-Transformation	diskrétní Furierova transformace
der Filter mit endlicher Impulsantwort	filtr s konečnou impulsní charakteristikou
der Filter mit unendlicher Impulsantwort	filtr s nekonečnou impulsní charakteristikou
der Filter	filtr
die Fourier-Transformation	Furierova transformace
der Frequenzgang	frekvenční charakteristika
das Frequenzspektrum	frekvenční spektrum
die Grenzfrequenz	mezní kmitočet
der Idealfilter	ideální filtr
der Kammfilter	hřebenový filtr
die Kantenglättung	vyhlazovací filtr
der lineare Filter	lineární filtr
der nichtlineare Filter	nelineární filtr
das Nyquist-Shannonsche Abtasttheorem	Nyquistovo-Shannonovo kritérium
die Ordnung des Filters	řád filtru
der passive Filter	pasivní filtr
der Phasengang	fázová charakteristika
der Tiefpassfilter	dolní propust
der Tschebyschefffilter	Čebyševův filtr
die Übertragungsfunktion	přenosová funkce
das Ursprungssignal	původní, vstupní signál
die Z-Transformation	Z-transformace

6.1.2 Abtastung

der Abtaster	vzorkovač
die Abtastung	vzorkování
das Abtasttheorem	vzorkovací teorém
die Digitalisierung	digitalizace
die diskrete Abtastung	diskrétní vzorkování
die sinc-Funktion	funkce sinc

die Untertaktung
das Ursprungssignal

podvzorkování
původní, vstupní signál

6.1.3 Quantisierung

das Digitalsignal
der diskrete Wert
die logarithmische Quantisierung
das Originalsignal
die Puls-Code-Modulation
die Quantisierung
der Quantisierungsfehler
das Quantisierungsgeräusch
die Quantisierungskennlinie
das Quantisierungsrauschen

digitální signál
diskrétní hodnota
logaritmické kvantování
originální signál
pulsní kódová modulace
kvantování
chyba kvantování
kvantizační šum
kvantizační hladiny
kvantovací šum

6.1.4 Analog-Digital-Wandler

der Abtaster
die Abtastfrequenz
die Amplitude
der Analog-Digital-Wandler
die Auflösung
der Augenblickswert
der BCD-Code
der binärer Code
die Bipolartechnik
der Datenstrom
die Delta-Sigma-Modulation
der Delta-Sigma-Wandler
die digitale Daten
der dynamische Fehler
der Fehler der Verstärkerung
die Flip-Flop-Schaltung
der Gray-Code
das höchstwertige Bit
der Integrator
der Integrierkomparator
das Jitter
der Komparatorprinzip
der Kompensationprinzip
die Konvertierungsgeschwindigkeit
der Näherungsprinzip
das niedrigstwertige Bit
der numerische Code
der Parallelumsetzer
der Quarzoszillator
die Referenzgröße
der Register
die Rückführschaltung

vzorkovač
vzorkovací kmitočet
amplituda
A/D převodník
rozlišení převodníku
okamžitá hodnota
BCD kód
binární kód
bipolární technologie
datový tok
delta-sigma modulace
delta-sigma komparátor
digitální data
dynamická chyba
chyba zesílení
klopný obvod
Grayův kód
nejvíce významný bit
integrátor
integrační komparátor
časová nejistota
komparační princip
kompenzační princip
rychlost převodu
aproximační princip
nejméně významný bit
číselný kód
paralelní převodník
Quartz oscilátor
referenční hodnota
registr
zpětná vazba

der Sägezahngenerator
 das Schaltlaufwerk
 der Schaltungsaufwand
 der Schaltungsaufwand
 der Taktimpuls
 temperaturabhängig
 der Übersetzungsfehler
 die Umrechnungscharakteristik
 die Umsetzungsgeschwindigkeit
 der Umsetzungsvorgang
 die Umsetzungszeit
 die Unipolartechnik
 die Unterscheidungsvermögen
 der Zähler
 das Zählergebnis
 der Zählverfahren Wandler
 die Zweitaktintegration

generátor pilového průběhu
 řídicí jednotka
 obvodové zapojení
 levný obvod
 hodinový signál
 teplotně závislý
 chyba převodu
 převodní charakteristika
 přenosová rychlost
 přeložené slovo
 převodová rychlost
 unipolární technologie
 rozlišovací schopnost
 čítač
 výstupní slovo
 sériový převodník
 dvoutaktní integrace

6.1.5 Digital-Analog-Wandler

die alternative Linaercharakteristik
 das analoge Ausgangssignal
 die Auflösungsvermögen
 der Decoder
 der Digital-Analog-Wandler
 das digitale Eingangssignal
 der effektive Anzahl von Bits
 der gekette Stand
 der getrennte Stand
 die ideale Charakteristik
 die Nachwirkung
 die Nichtlinearität
 der Nichtlinearitätsfehler
 der Nullpunktfehler
 das Mehrrampenverfahren
 der Operationsverstärker
 die Quelleinhaltung
 das R2R-Netzwerk
 der statische Fehler
 das Steuerglied
 die Überschaltzeit
 das Widerstandnetzwerk
 die wirkliche Charakteristik

náhradní přímková charakteristika
 analogové výstupní napětí
 rozlišovací schopnost
 dekodér
 Č/A převodník
 digitální vstupní signál
 efektivní počet bitů
 sepnutý stav
 rozepnutý stav
 ideální charakteristika
 hystereze
 nelinearita
 chyba nelinearity
 chyba nuly
 váhový dekodér
 operační zesilovač
 stabilita zdroje
 síť R-2R
 statická chyba
 budič
 doba přepínání
 odporová síť
 skutečná charakteristika

6.1.6 Spektralanalyse

die Amplitudenspektrum
 die Oberwelle
 der Spektralanalysator
 die Spektralanalyse

amplitudové spektrum
 harmonická
 spektrální analyzátor
 spektrální analýza

die Spektrallinie
das Spektrum

spektrální čára
spektrum

6.2 Tschechisch-Deutsch Wörterbuch

6.2.1 Antialiasing

aktivní filtr	der aktive Filter
amplitudová charakteristika	der Amplitudengang
analogový filtr	der analoge Filter
analogový signál	der Analo signal
Butterworthfiltr	der Butterworthfilter
Čebyševův filtr	der Tschebyschefffilter
diferenciální rovnice	die Differentialgleichung
digitální filtr	der digitale Filter
diracův impuls	der Dirac-Impuls
diskrétní Furierova transformace	die diskrete Fourier-Transformation
dolní propust	der Tiefpassfilter
fázová charakteristika	der Phasengang
filtr	der Filter
filtr s konečnou impulsní charakteristikou	der Filter mit endlicher Impulsantwort
filtr s nekonečnou impulsní char.	der Filter mit unendlicher Impulsantwort
frekvenční charakteristika	der Frequenzgang
frekvenční spektrum	das Frequenzspektrum
Furierova transformace	die Fourier-Transformation
hřebenový filtr	der Kammfilter
ideální filt	der Idealfilter
lineární filtr	der lineare Filter
mezní kmitočet	die Grenzfrequenz
nelineární filtr	der nichtlineare Filter
Nyquistovo-Shannonovo kritérium	das Nyquist-Shannonsche Abtasttheorem
pasivní filtr	der passive Filter
pásmová propust	der Bandpassfilter
pásmová zádrž	der Bandstopfilter
přenosová funkce	die Übertragungsfunktion
původní, vstupní signál	das Ursprungssignal
řád filtru	die Ordnung des Filters
šířka pásma	die Bandbreite
vyhlazovací filtr	die Kantenglättung
Z-transformace	die Z-Transformation

6.2.2 Vzorkování

digitalizace	die Digitalisierung
diskrétní vzorkování	die diskrete Abtastung
funkce sinc	die sinc-Funktion
podvzorkování	die Untertaktung
původní, vstupní signál	das Ursprungssignal
vzorkovací teorém	das Abtasttheorem
vzorkovač	der Abtaster

vzorkování

die Abtastung

6.2.3 Kvantování

digitální signál
diskrétní hodnota
chyba kvantování
kvantizační hladiny
kvantizační šum
kvantovací šum
kvantování
logaritmické kvantování
originální signál
pulsní kódová modulace

das Digitalsignal
der diskrete Wert
der Quantisierungsfehler
die Quantisierungskennlinie
das Quantisierungsgeräusch
das Quantisierungsrauschen
die Quantisierung
die logarithmische Quantisierung
das Originalsignal
die Puls-Code-Modulation

6.2.4 Analogově digitální převodníky

A/D převodník
amplituda
aproximační princip
BCD kód
binární kód
bipolární technologie
časová nejistota
číselný kód
čítač
datový tok
delta-sigma komparátor
delta-sigma modulace
digitální data
dvoutaktní integrace
dynamická chyba
generátor pilového průběhu
Grayův kód
hodinový signál
chyba převodu
integrační komparátor
integrátor
klopný obvod
komparační princip
kompenzační princip
levný obvod
nejméně významný bit
nejvíce významný bit
obvodové zapojení
okamžitá hodnota
paralelní převodník
přeložené slovo
přenosová rychlost
převodní charakteristika

der Analog-Digital-Wandler
die Amplitude
der Näherungsprinzip
der BCD-Code
der binärer Code
die Bipolartechnik
das Jitter
der numerische Code
der Zähler
der Datenstrom
der Delta-Sigma-Wandler
die Delta-Sigma-Modulation
die digitale Daten
die Zweitaktintegration
der dynamische Fehler
der Sägezahn-generator
der Gray-Code
der Taktimpuls
der Übersetzungsfehler
der Integrierkomparator
der Integrator
die Flip-Flop-Schaltung
der Komparatorprinzip
der Kompensationprinzip
der Schaltungsaufwand
das niedrigstwertige Bit
das höchstwertige Bit
der Schaltungsaufwand
der Augenblickswert
der Parallelumsetzer
der Umsetzungsvorgang
die Umsetzungsgeschwindigkeit
die Umrechnungscharakteristik

převodová rychlost	die Umsetzungszeit
Qaurtz oscilátor	der Quarzoszillator
referenční hodnota	die Referenzgröße
registr	der Register
rozlišení převodníku	die Auflösung
rozlišovací schopnost	die Unterscheidungsvermögen
rychlost převodu	die Konvertierungsgeschwindigkeit
řídící jednotka	das Schaltlaufwerk
sériový převodník	der Zählverfahren Wandler
teplotně závislý	temperaturabhängig
unipolární technologie	die Unipolartechnik
výstupní slovo	das Zählergebnis
vzorkovací kmitočet	die Abtastfrequenz
vzorkovač	der Abtaster
zpětná vazba	die Rückführschaltung

6.2.5 Digitálně analogové převodníky

analogové výstupní napětí	das analoge Ausgangssignal
budič	das Steuerglied
Č/A převodník	der Digital-Analog-Wandler
dekodér	der Decoder
digitální vstupní signál	das digitale Eingangssignal
doba přepínání	die Überschaltzeit
efektivní počet bitů	der effektive Anzahl von Bits
hystereze	die Nachwirkung
chyba nelinearity	der Nichtlinearitätsfehler
chyba nuly	der Nullpunktfehler
chyba zesílení	der Fehler der Verstärkerung
ideální charakteristika	die ideale Charakteristik
náhradní přímková charakteristika	die alternative Linaercharakteristik
nelinearita	die Nichtlinearität
odporová síť	das Widerstandnetzwerk
operační zesilovač	der Operationsverstärker
rozepnutý stav	der getrennte Stand
rozlišovací schopnost	die Auflösungsvermögen
skutečná charakteristika	die wirkliche Charakteristik
sepnutý stav	der gekette Stand
síť R-2R	das R2R-Netzwerk
stabilita zdroje	die Quelleinhaltung
statická chyba	der statische Fehler
váhový dekodér	das Mehrrampenverfahren

6.2.6 Spekrální analýza

amplitudové spektrum	die Amplitudenspektrum
harmonická	die Oberwelle
spektrální analyzátor	der Spektralanalysator
spektrální analýza	die Spektralanalyse
spektrální čára	die Spektrallinie

spektrum

das Spektrum

6.3 Antialiasing

aktiver Filter

Der aktive Filter heißen wir den Filter, in dem ein aktives Element ist. Zum Beispiel ein Operationsverstärker.

Resultierende Übertragung ist ein Produkt der Übertragung des passiven Filters und der Übertragung des aktiven Elementes.

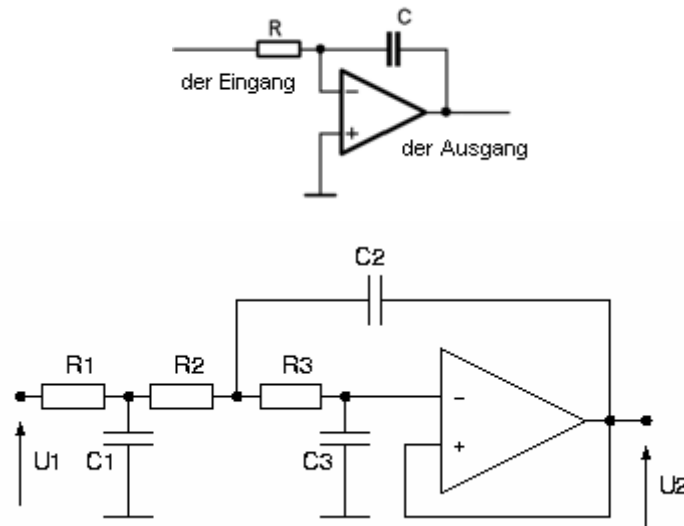


Bild. 6.3.1 Verschiedene aktive Filter

Amplitudengang

Der Amplitudengang gibt Übertragung des Elementes in die Abhängigkeit auf die Frequenz an. Ein Result haben wir in Dezibels oder ohne Einheit.

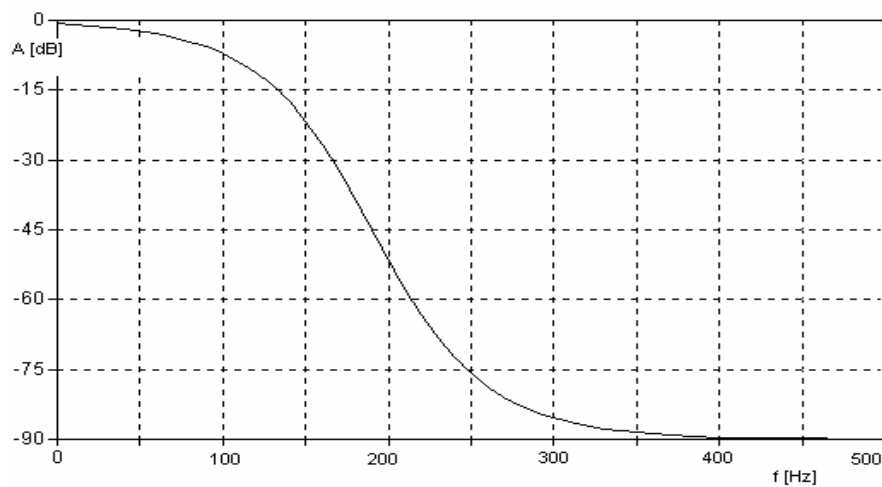


Bild 6.3.2 Beispiel des Amplitudenganges

analoger Filter

Den analogen Filter heißen wir ein Filter, der ein analoges Eingangssignal bearbeitet.

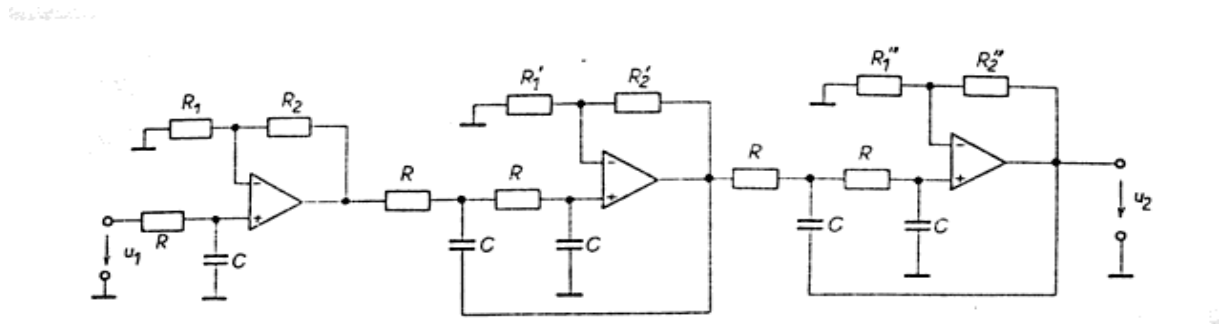


Bild 6.3.3 analoger Filter

Analogsignal

Das Analogsignal hat nicht endliche Zahl des Standes. Es ist zusammenhängend in der Zeit. Meistens denken wir unter dem Begriff „der Analogsignal“ die Sinuslinie mit verschiedenen Frequenzen, Phasen und Amplituden.

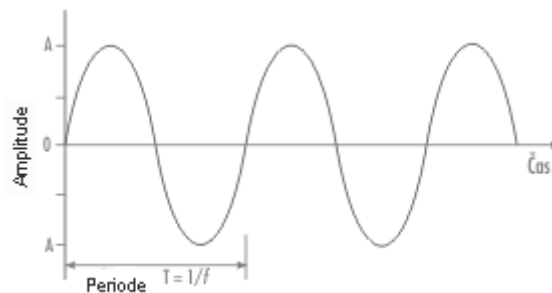


Bild 6.3.4 Analogsignal

Bandbreite

Die Bandbreite B [Hz] gibt die Differenz der Grenzfrequenzen des verdrängten oder des übertragenen Bandes an.

Die Grenzfrequenz des verdrängten und übertragenen Bandes ist das Absinken über 3 dB von der Top-Werte der Übertragung.

$$B = f_v - f_u \quad (6.3.1)$$

wo: B ...Bandbreite
 f_v ...verdrängte Band
 f_u ...übertragene Band

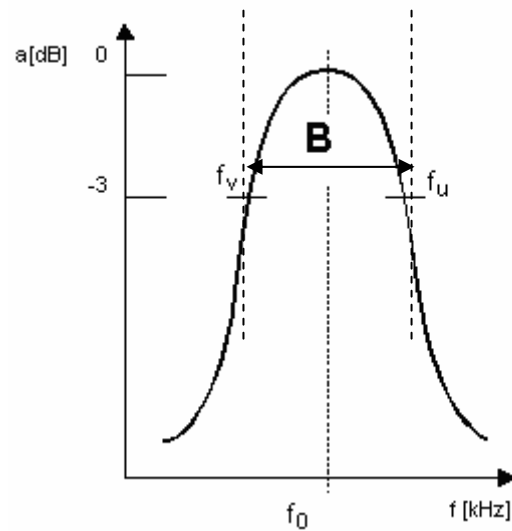
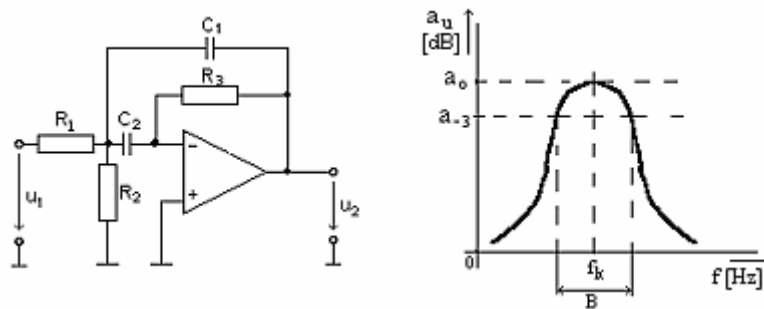


Bild 6.3.5 Bandbreite

Bandpassfilter

Der Bandpassfilter verhält alle frequenzen außer der Bandbreite.

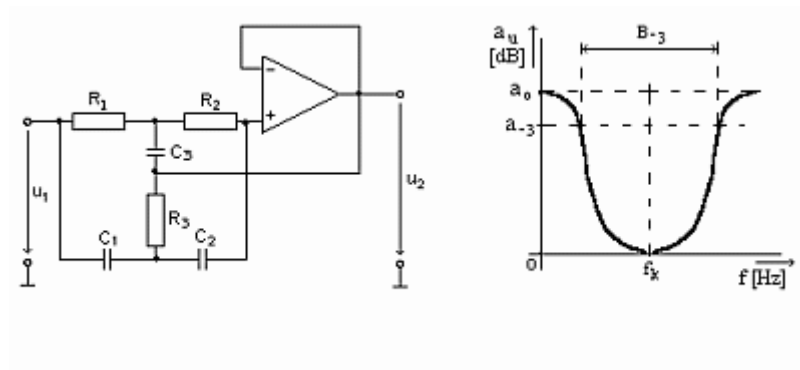


wo: B ...Bandbreite
 f_k ...Frequenz mit maximale Übertragung
 a_0 ...maximale Übertragung
 a_{-3} ...Absinken an 3 dB von maximale Übertragung

Bild 6.3.6 realer Bandpassfilter

Bandstopfilter

Der Bandpassfilter lässt allen frequenzen in der Bandbreite durch.



wo: B ...Bandbreite
 f_k ...Frequenz mit maximale Übertragung
 a_0 ...maximale Übertragung
 a_{-3} ...Absinken an 3 dB von maximale Übertragung

Bild 6.3.7 realer Bandstoppfiler

Butterworthfilter

Der Butterworthfilter ist die Sorte des Filters. Er hat glattern Amplitudengang und linearen Phasengang.

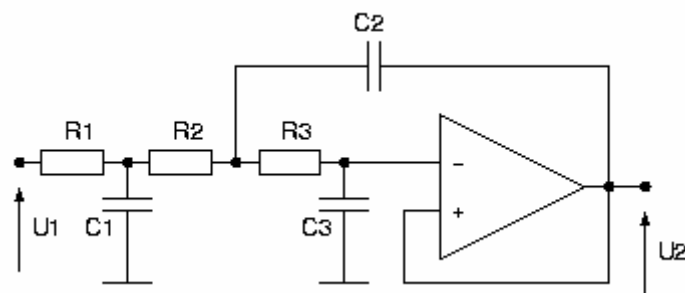


Bild 6.3.8 Butterworthfilter

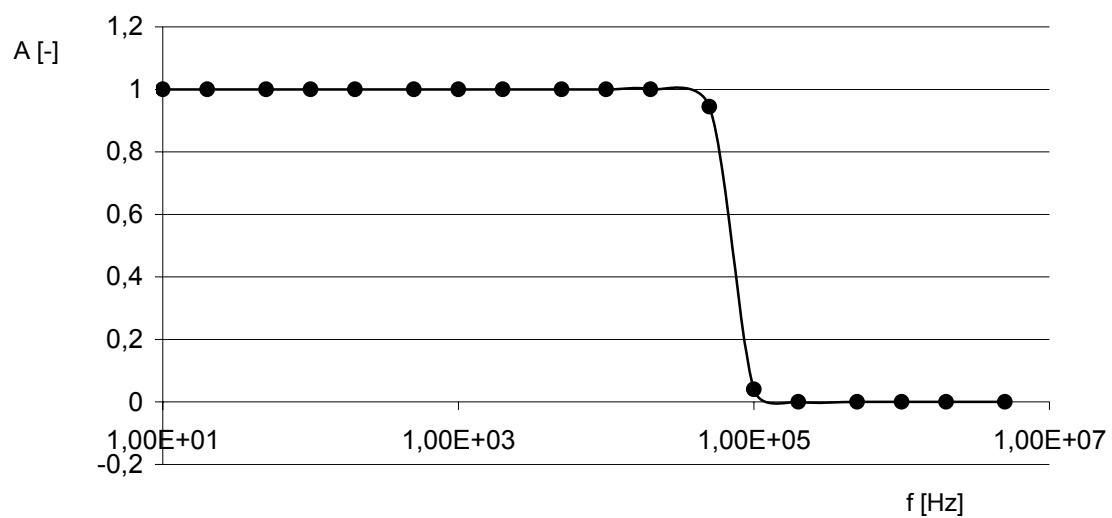


Bild 6.3.9 Amplitudengang des Butterworthfiltes

Differentialgleichung

Für die Rechnung des digitalen Filters brauchen wir die Differentialgleichung. Die Differentialgleichung kann man sein in der Form:

$$y^{(n)} + a_{n-1}(x)y^{(n-1)} + a_{n-2}(x)y^{(n-2)} + \dots + a_1(x)y' + a_0(x)y = b(x) \quad (6.3.2)$$

wo: $y^{(n)}$die Ordnung der Derivation
 $a_{n-1}(x)$...die differentiationsfunktion der Veränderliche x
 $b(x)$die Gesamtfunktion

digitaler Filter

Der digitale Filter ist ein Filter, der das digitale Signal bearbeitet.

Wir haben zwei Sorten des digitalen Filters: der Filter mit endlicher Impulsantwort (FIR)
der Filter mit unendlicher Impulsantwort (IIR)

Mit Hilfe der Rückführschaltung und des Verzögerungselementes können wir alle Sorten des Filters machen.

Dirac-Impuls

Der Dirac-Impuls ist ein Signal, das unendlich kurze Zeit und unendliche „Amplitude“ hat. Mit Hilfe des Dirac-Impulses drücken wir ein ideales Übernahmesignal.

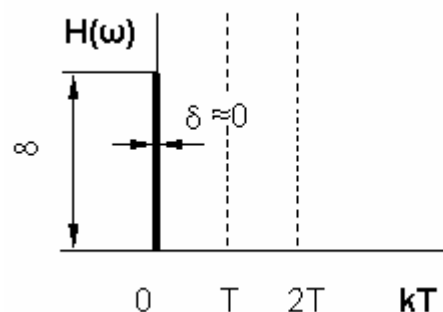


Bild 6.3.10 Dirac-Impuls

diskrete Fourier-Transformation

Mit Hilfe der Fourier-Transformation können wir das Spektrum des Signales ausdrücken. Die diskrete Fourier-Transformation drückt einen Bestandteil des Spektrums.

$$F(k\Omega) = \sum_{n=0}^{N-1} f(nT)e^{-jk\Omega nT} \quad (6.3.3)$$
$$f(nT) = \frac{1}{N} \sum_{k=0}^{N-1} F(k\Omega)e^{jk\Omega nT}$$

wo: $k\Omega$...Frequenzmuster
 nT ...Zeitpunkt der Muster
 N ...Zahl der Muster

Filter mit endlicher Impulsantwort (FIR)

Der Filter mit endlicher Impulsantwort ist ein digitaler Filter ohne die Rückführschaltung. Die Filter FIR sind absolut standsichere. Ihr Phasegang ist linearer, wenn sie symmetrisch oder antisymmetrisch sind. Ihre Realisation ist einfach.

Filter mit unendlicher Impulsantwort (IIR)

Der Filter mit endlicher Impulsantwort ist ein digitaler Filter mit der Rückführschaltung. Die Filter IIR haben mehrere Variante der Tätigkeiten. Sie können linear oder nichtlinear sein. Seine Realisation ist ganz schwer und man kann passieren, dass der ausgeführte Filter IIR unbeständig wird.

Filter

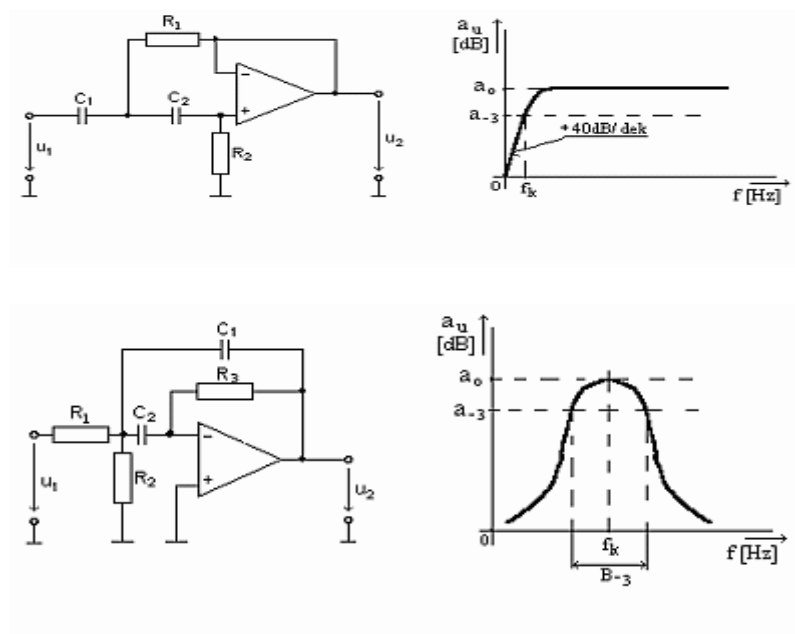
Der Filter kann einige Frequenzen des Spektrums verhalten.

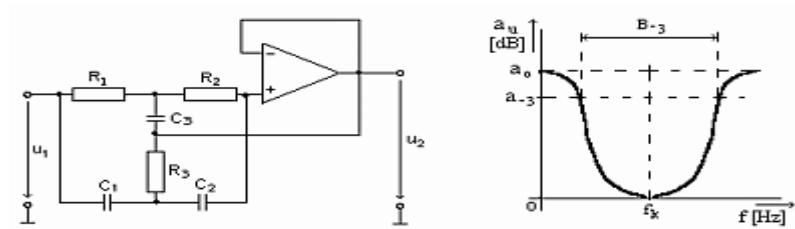
Die Filter sind:

- digitale $\begin{cases} \text{FIR} \\ \text{IIR} \end{cases}$
- analoge $\begin{cases} \text{passive} \\ \text{aktive} \end{cases}$

Gattungsmäßig der Filtrierung teilen wir die Filter auf den Tiefpassfilter, den Hochpassfilter den Bandpassfilter und den Bandstopfilter an.

Der filter beschreibt sein Frequenzgang oder seine Übertragungsfunktion





wo: B ...Bandbreite
 f_k ...Frequenz mit maximale Übertragung
 a_0 ...maximale Übertragung
 a_{-3} ...Absinken an 3 dB von maximale Übertragung

Bild 6.3.11 verschiedene Filter

Fourier-Transformation

Mit Hilfe der Fourier-Transformation können wir das Spektrum des Signales ausdrücken.

$$F(\omega) = \sum_{-\infty}^{\infty} f_n e^{-j\omega nT} \quad (6.3.4)$$

wo: $F(\omega)$...Größe der Muster
 f_n ...Frequenz der Muster
 nT ... Zeitpunkt der Muster

Frequenzgang

Der Frequenzgang des Filters hat zwei Teile, den Amplitudengang, der die Übertragung des Filters angibt, und der Phasegang, die Verspätung des Ausgangs zum Eingang angibt.

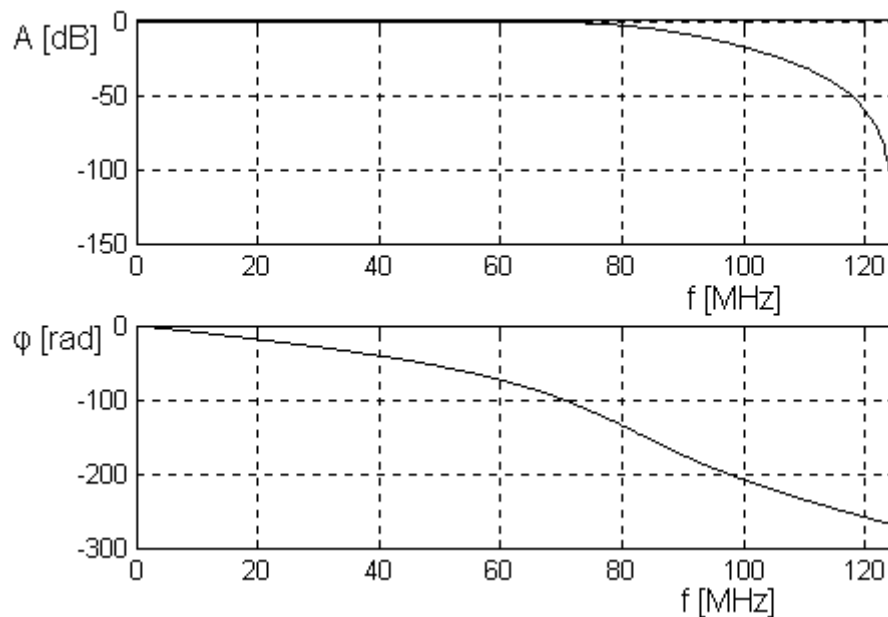


Bild 6.3.12 Frequenzgang

Frequenzspektrum

Jedes Signal ist zusammengesetzt aus Oberwellen. Jede Oberwelle hat ihre Frequenz, Amplitude und Phase. Das Frequenzspektrum gibt die Bedeutung und die Verspätung der einzelnen Oberwellen des Signals an.

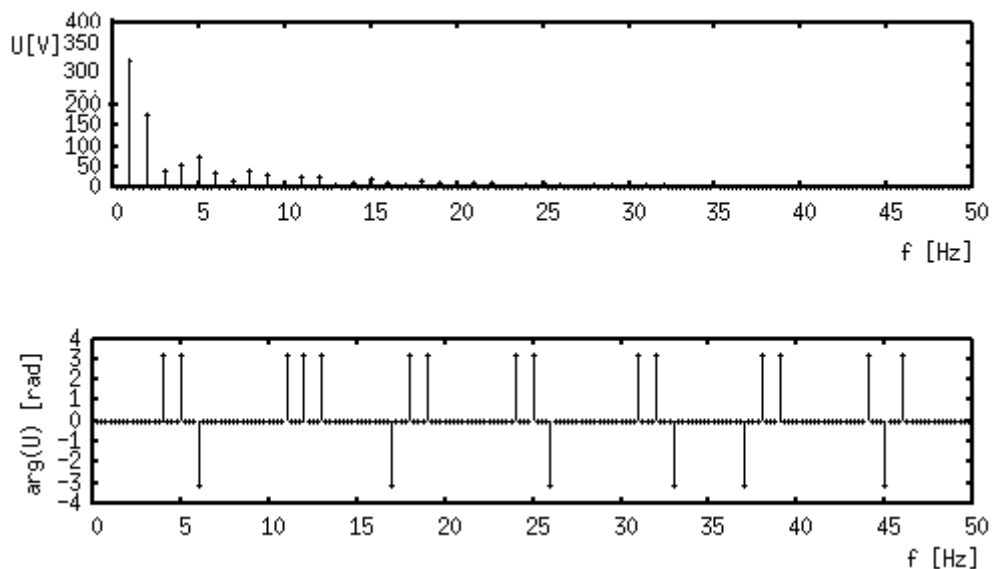


Bild 6.3.13 Frequenzspektrum

Grenzfrequenz

Die Grenzfrequenz f_g des Filters ist die Frequenz, bei der die Übertragung des Filters um 3 dB gegenüber größter Übertragung des Filters fällt

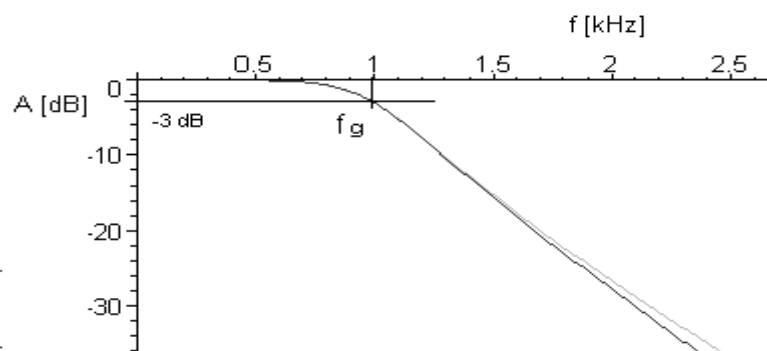


Bild 6.3.14 Grenzfrequenz

Idealfilter

Der Idealfilter ist ein Filter, der idealen Frequenzgang hat. Das bedeutet, dass in der Bandbreite die Übertragung 1 und dann 0 ist.
In der Praxis dieser Filter existiert nicht.

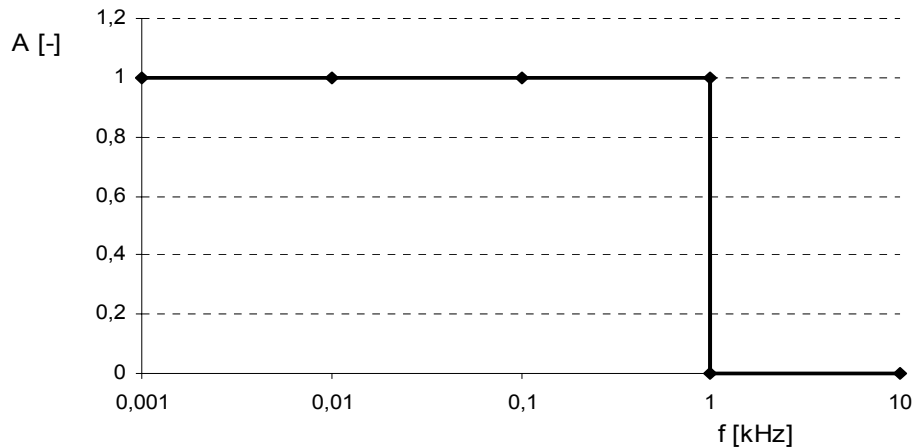


Bild 6.3.15 Tiefpassfilter als der Idealfilter

Kammfilter

Der Kammfilter ist eine Sorte des digitalen Filters mit endlicher Impulsantwort. Dieser Filter kann man als einen Tiefpassfilter benutzen.

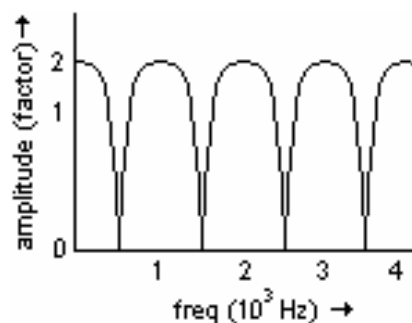


Bild 6.3.16 Frequenzgang des Kammfilters

Kantenglättung

Das Signal, das aus Digital-Analog-Wandler geht, ist nicht zusammenhängend, aber es hat die Staffelnungen. Für Verundung des Signals müssen wir die Kantenglättung benutzen.

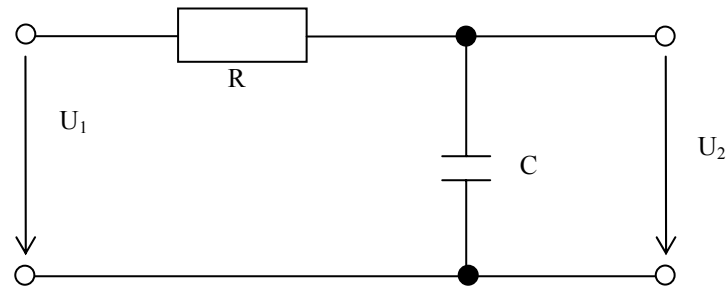


Bild 6.3.17 Kantenglättung

linearer Filter

Der lineare Filter ist ein Filter, der linearen Phasengang hat.

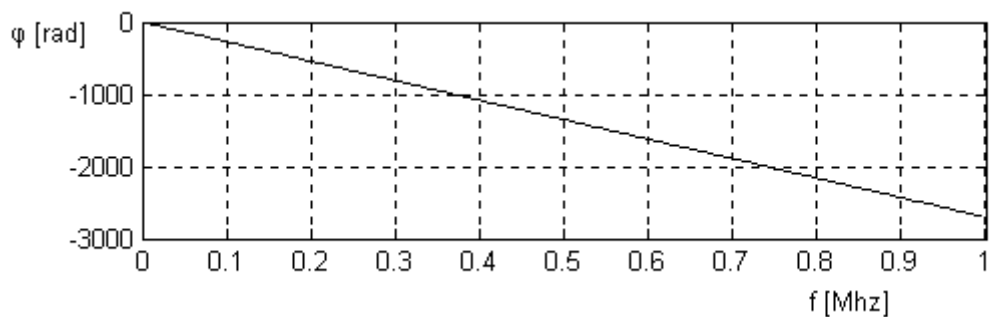


Bild 6.3.18 Phasengang des linearen Filters

nichtlinearer Filter

Der nichtlineare Filter ist ein Filter, der nichtlinearen Phasengang hat.

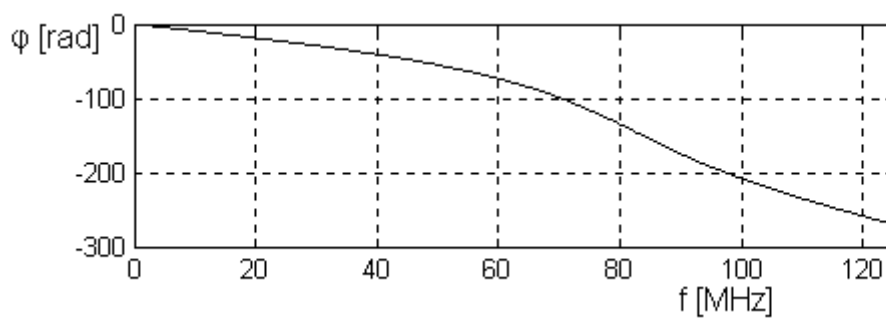


Bild 6.3.19 Phasengang des nichtlinearen Filters

Nyquist-Shannonsche Abtasttheorem

Dieses Theorem sagt, welche kleinste Frequenz man anwenden muss, damit man nicht zu der Untertaktung kommt.

$$f_a \geq 2 \cdot f_{\max} \quad (6.3.5)$$

wo: f_aAbtastfrequenz
 f_{max} ...höchste Frequenz des Signales

Die Abtastfrequenz ist Grenzfrequenz für den Antialiasingfilter.

Ordnung des Filters

Die Vereinigung des n-Filters erster Ordnung entsteht ein Filter n-ter Ordnung.
 Die Filter höherer Ordnung haben steileren Amplitudengang. Das bedeutet, dass der Filter bessere Eigenschaften haben wird.

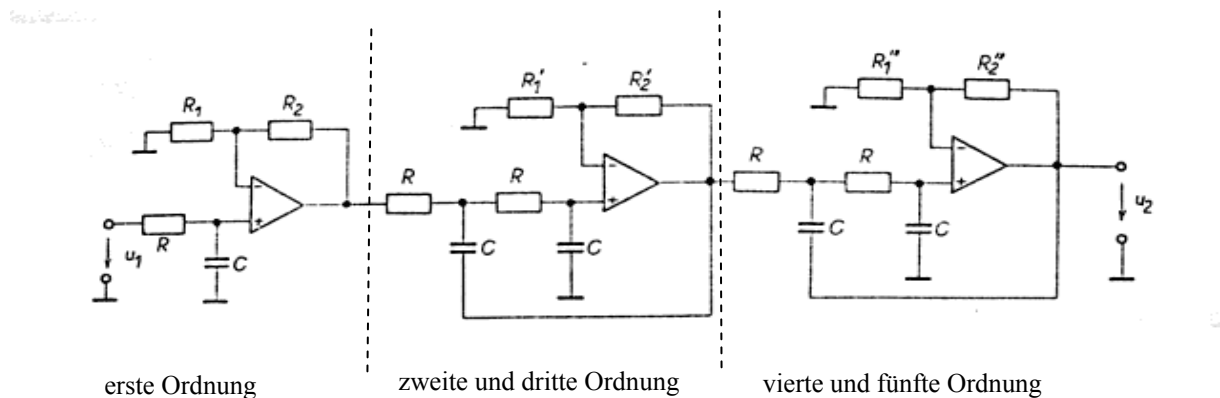


Bild 6.3.20 Ordnung des Filters

passiver Filter

Der passive Filter heißen wir den Filter, in dem kein aktives Element ist.

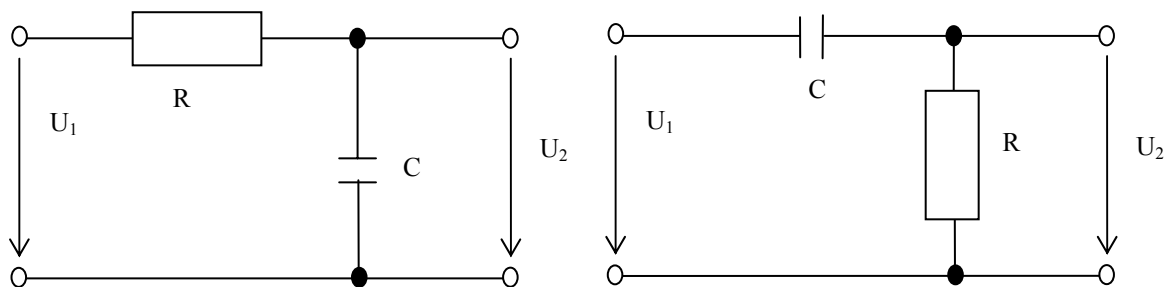


Bild 6.3.21 passiver Filter

Phasengang

Der Phasengang des Filters gibt die Verspätung einzelner Frequenzen auf dem Ausgang in Bezug auf dem Eingang an.

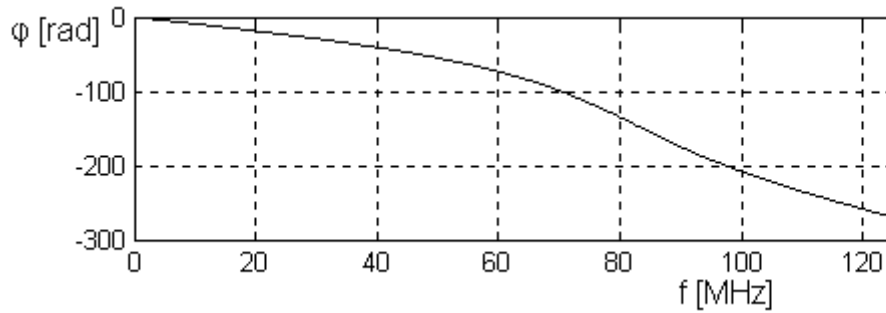


Bild 6.3.22 Phasengang

Tiefpassfilter

Der Tiefpassfilter (oder das Integrationsglied) wird als der Antialiasingfilter und die Kantenglättung gebraucht.

Für kleine Frequenzen hat er die Übertragung 1, für große Frequenzen hat er die Übertragung 0.

$$f_g = \frac{1}{2\pi RC} \quad (6.3.6)$$

wo: f_g ... Grenzfrequenz des RC-Filters
 R ... Widerstand des Resistor
 C ... Kapazität des Kondensator

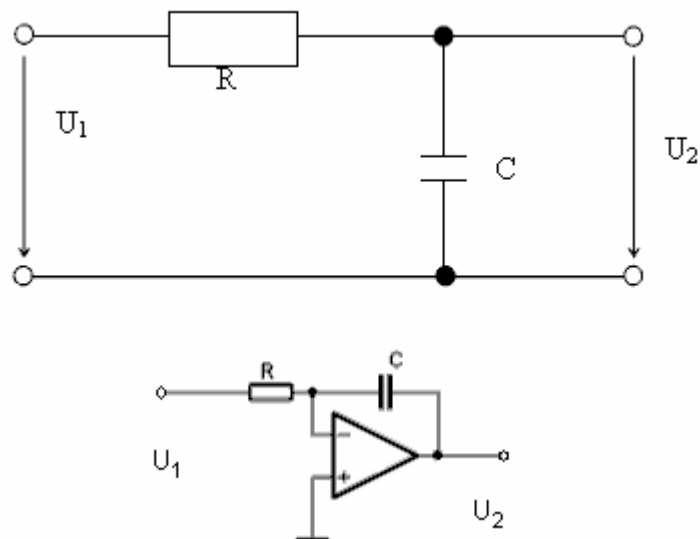


Bild 6.3.23 verschiedene Tiefpassfilter

Tschebyschefffilter

Der Tschebyschefffilter ist die Sorte des Filters. Er hat welliger Amplitudengang und linearer Phasengang.

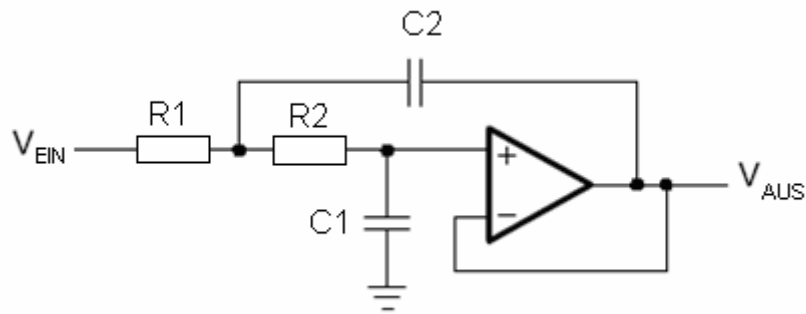


Bild 6.3.24 Tschebyschefffilter

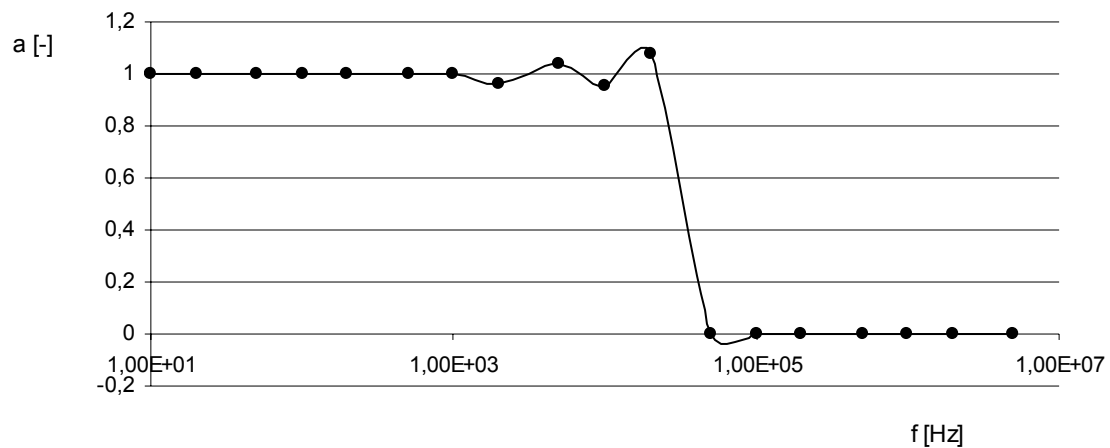


Bild 6.3.25 Amplitudengang des Tschebyschefffilters erste Variante

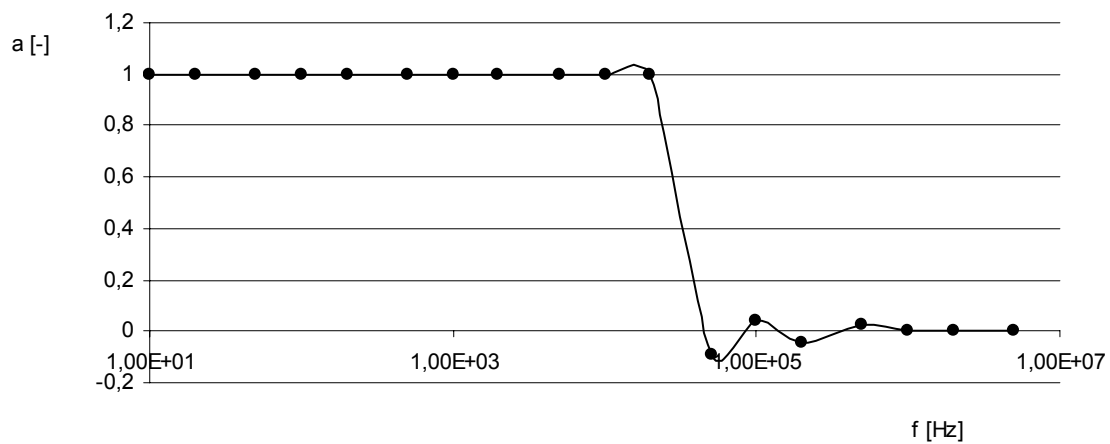


Bild 6.3.26 Amplitudengang des Tschebyschefffilters zweite Variante

Übertragungsfunktion

Die Übertragungsfunktion gibt das Verhältnis zwischen dem Ausgang und dem Eingang des Filters in ein Frequenzspektrum an. Die Übertragungsfunktion des Tiefpassfilters kann man aussehen:

$$A = \frac{1}{1 + sCR} \quad (6.3.7)$$

wo: A ...Übertragung des Tiefpassfilters
 s ...Parameter
 R ...Widerstand des Resistors
 C ...Kapazität des Kondensators

Ursprungssignal

Das Ursprungssignal ist ein Signal, das aus dem nützlichen Signal und aus dem Rausch geformt ist. Dieses Signal können wir mit dem Filter säubern.

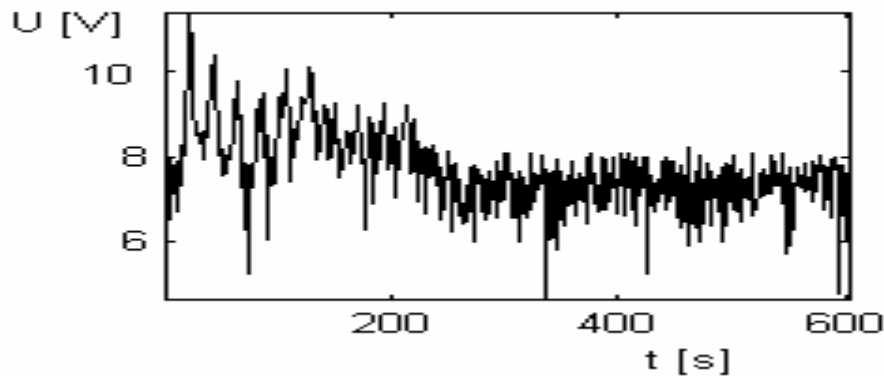


Bild 6.3.27 Beispiel des Ursprungssignals

Z-Transformation

Die Z-Transformation ist Komplexfunktion in der Form:

$$Z\{x_n\} = X(z) = \sum_{n=-\infty}^{\infty} x_n \cdot z^{-n} \quad (6.3.8)$$

wo: $X(z)$...Komplexfunktion
 nIndex der Probe
 x_nGröße der Probe
 zkomplexe Veränderliche

Mit Hilfe der Z-Transformation können wir digitalen Filter projektieren.

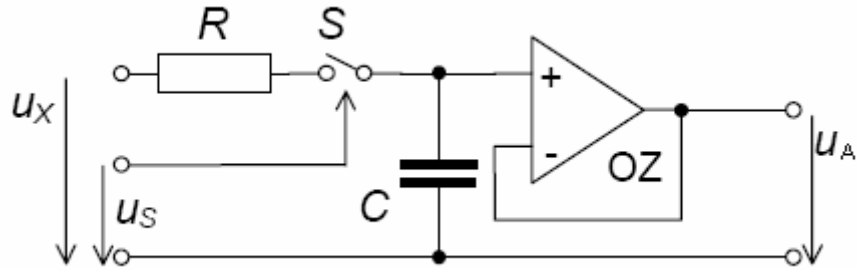
6.4 Analog-Digital-Wandler

6.4.1 Abtastung

Abtaster

Der Abtaster ist die elektronische Schaltung, die fähig ist, den Wert des Analogsignals zu behalten.

Der Schalter S schaltet sich mit der Abtastfrequenz. Der Kondensator C wird aufgeladen und dann mit Hilfe des Operationsverstärker wird der Wert der Probe auf den Ausgang übertragen.



wo: u_X ...Ursprungssignal
 u_S ...Tastzeit
 u_A ...eigentliche Probe

Bild 6.4.1 Abtaster

Abtastung

Die Abtastung ist erster Schritt bei der Digitalisierung. Mit Hilfe der Abtastfrequenz wird das Eingangssignal auf die Einzelteile, die Proben bemustert.

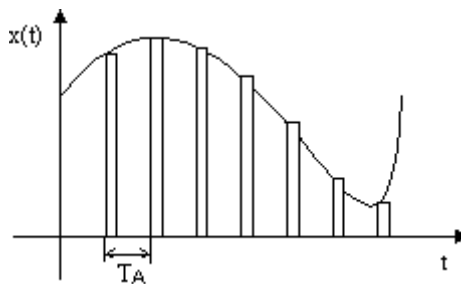


Bild 6.4.2 Abtastung

Abtasttheorem

Dieses Theorem sagt, welche kleinste Frequenz man anwenden muss, damit es nicht zu der Untertaktung kommt.

$$f_a \geq 2 \cdot f_{\max} \quad (6.4.1)$$

wo: f_aAbtastfrequenz
 $f_{\max}$...höchste Frequenz des Signales

Digitalisierung

Die Digitalisierung des Signals ist der Vorgang, bei dem sich Analogsignal auf Digitalsignal ändert.

Das Analogsignal muss man abtasten, quantisieren und in binär Code übertragen.

Bei der Digitalisierung begehen wir verschiedenen Fehler, aber mit Digitalsignal kann man besser arbeiten.

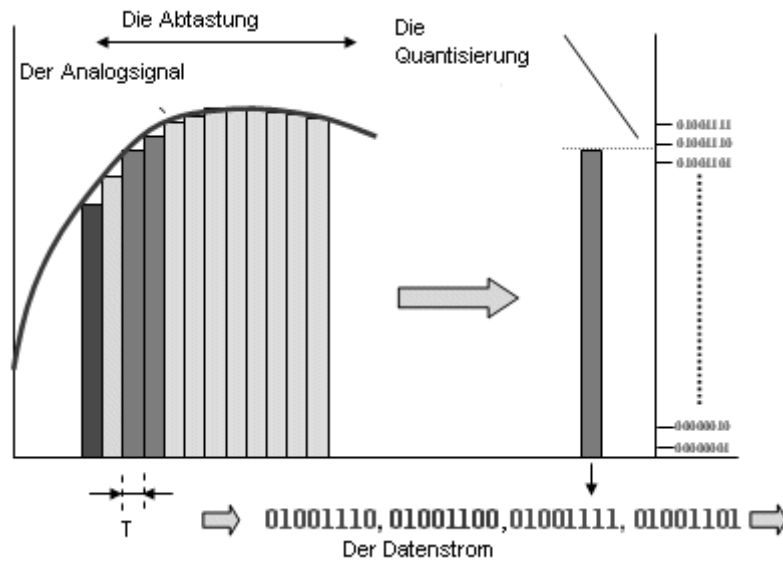


Bild 6.4.3 Digitalisierung

diskrete Abtastung

Die Abtastung ist erster Schritt bei der Digitalisierung. Mit Hilfe der Abtastfrequenz bemustert man das Eingangssignal auf die Einzelteile, die Proben. Nach der Abtastung bekommen wir diskrete Proben, dass das bedeutet wir über diskrete Abtastung sprechen.

sinc-Funktion

Die sinc-Funktion wird für die Abbildung des Spektrums des periodischen Signals gebraucht. Die sinc-Funktion hat die Form:

$$\text{sinc}(x) = \begin{cases} \frac{\sin x}{x} & , x \neq 0 \\ \text{und} \\ 1 & , x = 0 \end{cases} \quad (6.4.2)$$

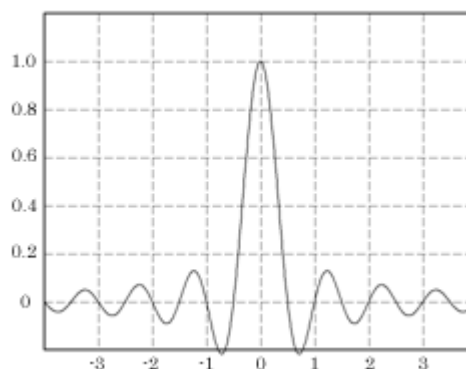


Bild 6.4.4 sinc-Funktion

Untertaktung

Wenn wir nicht das Abtasttheorem einhalten, blenden die Spektrums des Signal ein. Das veranlässt große Fehler. Deshalb gebrauchen wir Antialiasingfilter.

5.4.2 Quantisierung

Digitalsignal

Das Digitalsignal ist endliches Ergebnis des Analog-Digital-Wandlers. Quantisierte Werte übertragen den Coder auf das Digitalsignal.

Die Sorte des Digitalsignales kann man verschieden sein. Es hängt von der Sorte des Coders ab.

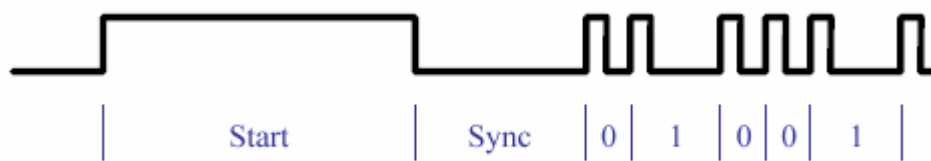


Bild 6.4.5 Digitalsignal

diskreter Wert

Jeder diskrete Wert, der mit der Abtastung entsteht, hat seine Größe.

Bei der Digitalization können wir nicht unendliche Zahl diskreter Werte bürden, deshalb müssen wir diese diskrete Werte auf die Größe der Quantisierungskennlinien quantisieren.

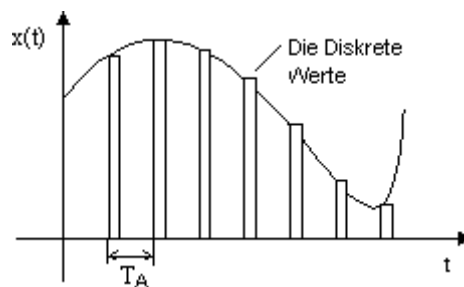


Bild 6.4.6 diskreter Wert

logarithmische Quantisierung

Manchmal brauchen wir logarithmisch quantisieren, deshalb gebrauchen wir logarithmische Quantisierungskennlinie.

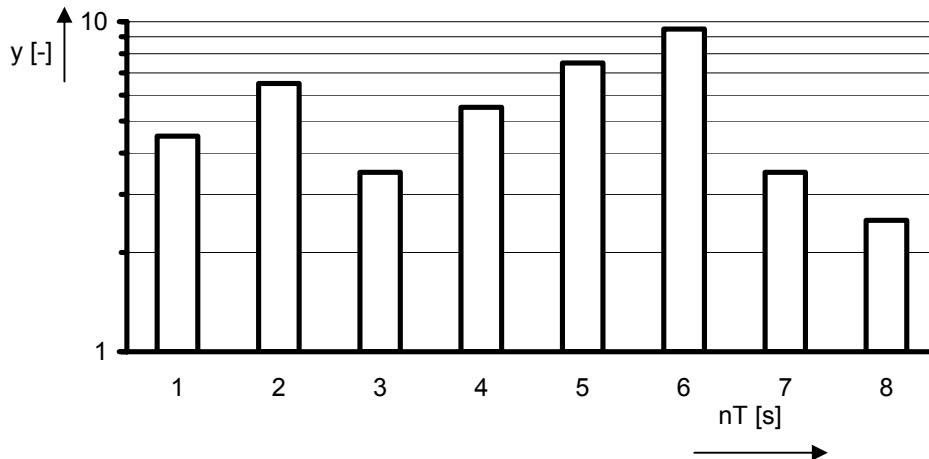


Bild 6.4.7 logarithmische Quantisierung

Originalsignal

Das Originalsignal ist ein Signal, das wir auf den Eingang des Analog-Digital-Wandlers zuleiten. Nach der Dekodierung möchten wir dieses Signal gleich haben.

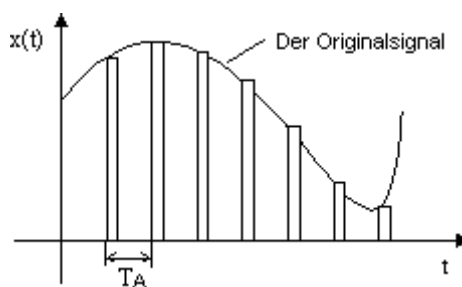


Bild 6.4.8 Originalsignal

Puls-Code-Modulation (PCM)

Die quantisierte Werte können wir auf PCM übertragen. Jede Quantisierungskennlinie hat seinen binären Code. Gemäß der Quantisierungskennlinie hat der Puls seine Breite und Form. Diese Übertragung macht Coder.

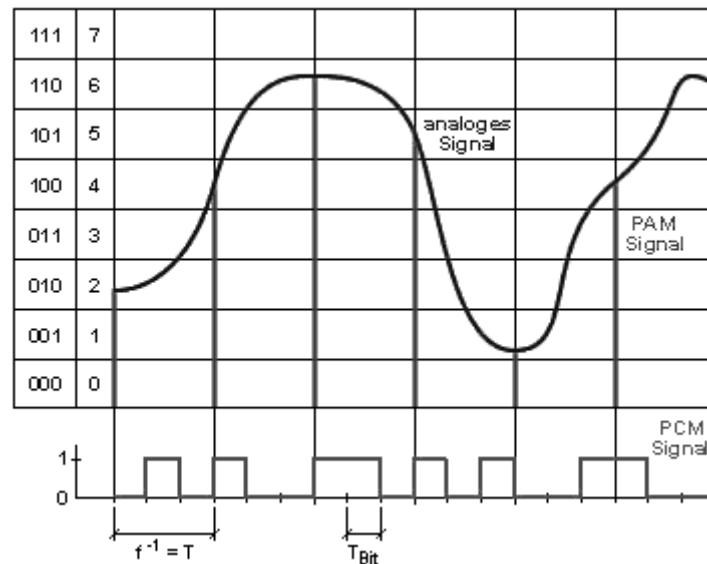


Bild 6.4.9 Puls-Code-Modulation

Quantisierung

Jeder diskreter Wert, der mit der Abtastung entsteht, hat seine Größe. Bei der Digitalization können wir nicht unendliche Zahl diskreter Werte bürden, deshalb müssen wir diese diskrete Werte auf die Größe der Quantisierungskennlinien quantisieren. Dieser Fehler heißt der Quantisierungsgeräusch. Der Abstand zwischen den Quantisierungskennlinien heißt Quantisierungsrauschen.

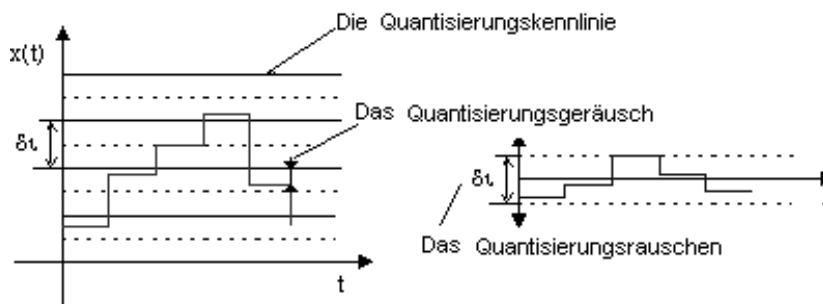


Bild 6.4.10 Quantisierung

Quantisierungsgeräusch

Das Quantisierungsgeräusch entsteht durch die Abrundung der diskreten Werte. Es ist die Differenz zwischen der Quantisierungskennlinie und wirklichem Wert.

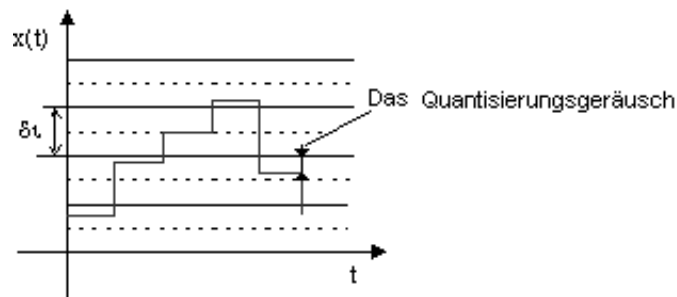


Bild 6.4.11 Quantisierungsgeräusch

Quantisierungskennlinie

Bei der Digitalization können wir nicht unendliche Zahl diskreter Werte bürden, deshalb müssen wir diese diskrete Werte auf die Größe der Quantisierungskennlinien quatisieren. Die Zahl der Bits gibt uns dei Zahl der Quantisierungskennlinien.

$$x = 2^n \quad (6.4.3)$$

wo: x ...Zahl der Quantisierungskennlinien
 n ...Zahl der Bits

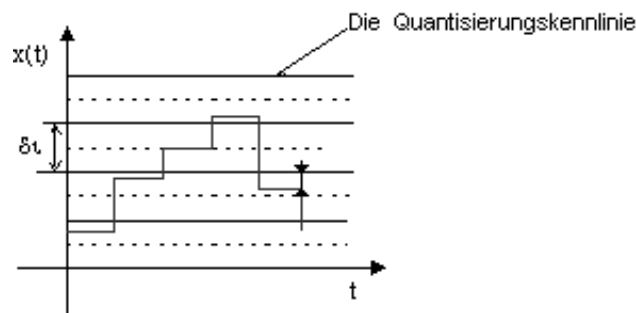


Bild 6.4.12 Quantisierungskennlinie

Quantisierungsrauschen

Es ist eigentlich die Diferenz zwischen den Quantisierungskennlinien. Es heißt auch Quantisierungsschritt q. Sein Wert ist der Wert des LSB.

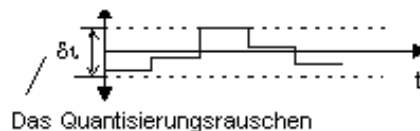


Bild 6.4.13 Quantisierungsrauschen

Vergleichsschaltung

Die Vergleichsschaltung ist die elektronische Schaltung, die den Wert des Eingangssignals mit der Referenzspannung abgleicht. Wenn der Wert gleich ist, ist auf dem Ausgang der

Vergleichsschaltung logische 1 (U_{cc}). Wir können die Kaskade mit verschiedener Referenzspannung machen und dann die Quantisierungskennlinien aufstellen.

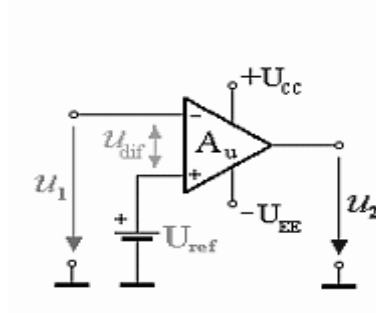


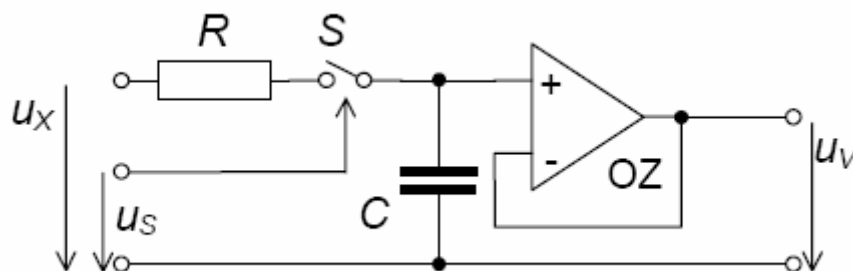
Bild 6.4.14 Vergleichsschaltung

6.4.3 Analog-Digital-Wandler

Abtaster

Der Abtaster ist die elektronische Schaltung, die fähig ist, den Wert des Analogsignals zu behalten.

Der Schalter S schaltet sich mit der Abtastfrequenz. Der Kondensator C wird aufgeladen und dann mit Hilfe des Operationsverstärker wird der Wert der Probe auf den Ausgang übertragen.



wo: u_x ...Ursprungssignal
 u_s ...Tastzeit
 u_v ...eigentliche Probe

Bild 6.4.15 Abtaster

Abtastfrequenz

Die Abtastfrequenz ist die Frequenz, mit der wir das Eingangssignal abtasten. Ihre Größe muss dem Nyquist-Shannonsche Abtasttheorems entsprechen.

Amplitude

Die Amplitude ist der größte Wert, den das Signal haben kann.

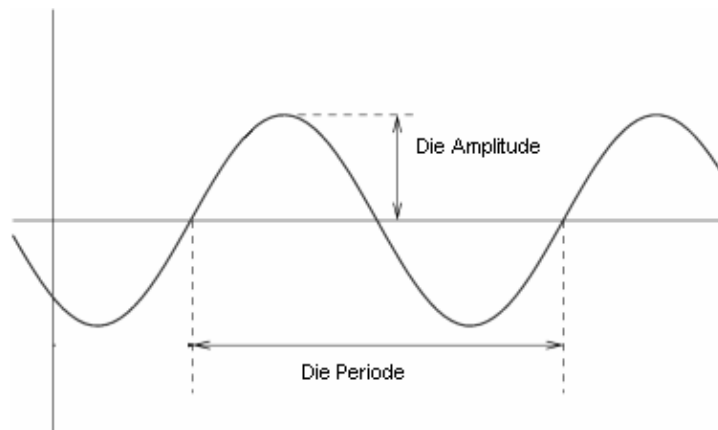


Bild 6.4.16 Amplitude

Analog-Digital-Wandler

Es ist das System, das aus verschiedenen Bauelementen zusammengebaut wird und das das Analogsignal auf das Digitalsignal überträgt.

Das Analogsignal verstärkt sich mit dem Verstärker. Der Antialiasingfilter dämpft die Frequenzen, die dem Nyquist-Shannonsche Abtasttheorem nicht entsprechen. Der Abtaster macht die Proben und Analog-Digital-Wandler (der Komparator) überträgt sie auf die Binärzahl. Der Koder verdatet die Binärzahl auf einiger Code für die Übertragung. Ganzes System ist mit der Taktimpuls gesteuert.

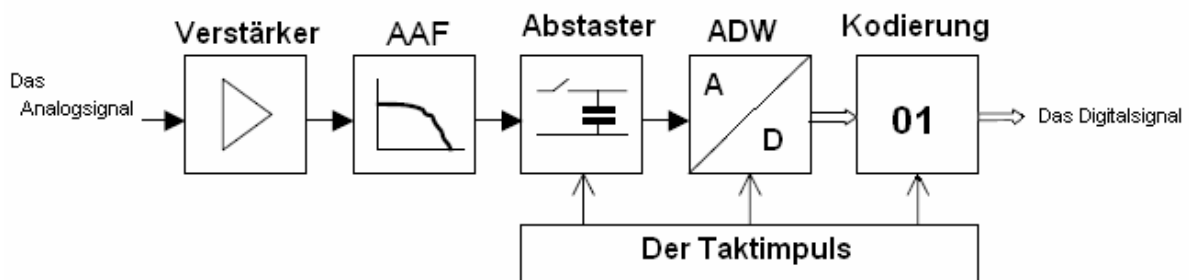


Bild 6.4.17 ganzer Analog-Digital-Wandler

Auflösung

Die Auflösung des Analog-Digital-Wandlers ist der Spannungswert des LSB oder auch der Größe der letzten Quantisierungskennlinie. Die Auflösung wächst mit der Zahl der Bits des Analog-Digital-Wandlers.

Augenblickswert

Das Analogsignal ist veränderlich in der Zeit. Der Augenblickswert ist der Wert des Signals in einem Augenblick.

$$u = U_{\max} \cdot (\sin 2\pi ft + \varphi) \quad (6.4.4)$$

wo: uAugenblickswert der Spannung
 $U_{\max}$...Amplitude der Spannung

fFrequenz der Spannung
 φPhase der Spannung

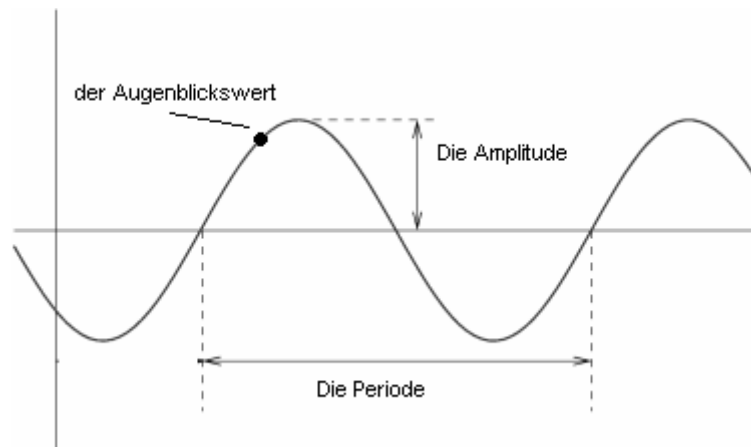


Bild 6.4.18 Augenblickswert

BCD-Code

Der BCD-Code ist die Kürzung für binär kodierte Dezimalsystem. Jede Ziffer hat ihre Kombination aus vier Bits zusammengestellt.
 Vier Bits können sechzehn Kombinationen geben. Wir brauchen nur zehn Kombinationen.
 Übriggeblieben Kombinationen heißen die Redundanz.

dekadische Wert	0	1	2	3	4	5	6	7	8	9	Die Redundanz
BCD-Code	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	

Tabelle 6.4.1 BCD - Code

binärer Code

Der binärer Code ist der Code, der dekadischen Wert auf die Binärzahl überträgt.

dekadischen Wert	binärer Code
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
10	1010
11	1011

12	1100
13	1101
14	1110
15	1111

Tabelle 6.4.2 binärer Code

Bipolartechnik

Die Bipolartechnik gibt nur bipolare Elemente an.

Die Bipolartechnik ist teuer, sie braucht viele Energie und sie ist schneller als die Unipolartechnik. Versorgungsspannung des Komparator kann negativ sein und dann kann man sich negativ Analogsignal verarbeiten.

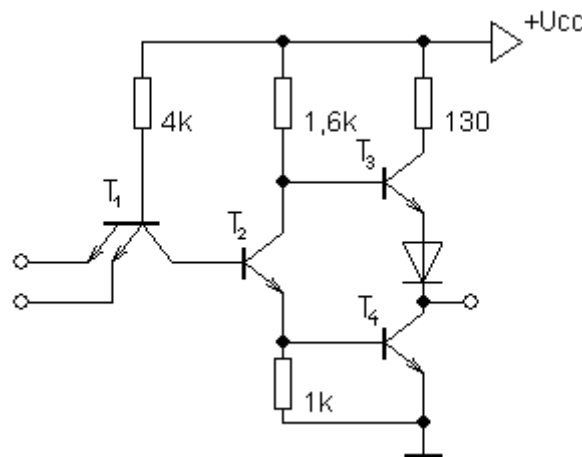


Bild 6.4.19 Elektronisches Tor NAND in der Bipolartechnik

Datenstrom

Auf dem Ausgang des Analog-Digital-Wandlers sind die Daten zu nächster Bearbeitung verschickt. Einzelne Wörter gestalten den Datenstrom.

Delta-Sigma-Modulation

Der Modulator beinhaltet den Integrier (den Tiefpassfilter), den Komparator K, die Flip-Flop-Schaltung des Types D, die sich mit dem Taktimpuls f_s umklappt. Rückführung mit dem Digital-Analog-Wandler schaltet die Versorgungsspannung auf die Werten $\pm U_R$ um, die sich von dem Eingangsspannung mit Hilfe des Operationsverstärker abnimmt.

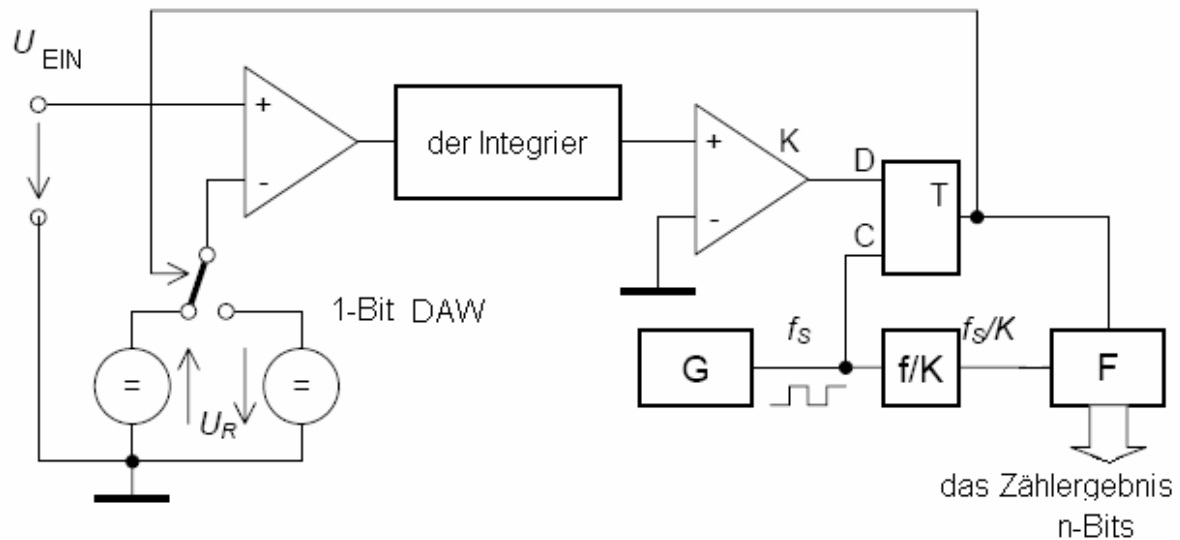


Bild 6.4.20 Delta-Sigma-Modulation

Delta-Sigma-Wandler

Der Delta-Sigma-Wandler ermöglicht hohe Linearität mit hoher Auflösung. Die Umsetzungszeit ist niedrig.

Bei diesem Wandler entsteht das Quantisierungsrauschen, dessen Spektrum im Intervall $\left\langle 0, \frac{f_s}{2} \right\rangle$ liegt. Weil bei diesem Wandler mit der Frequenz f_s bemustert wird, die K-mal das Abtasttheorem überschreitet wird, die Breite des Spektrums der Leistung des Quantisierungsrauschen K-mal ausgebreitet. Der Digitalfilter übt antialiasing.

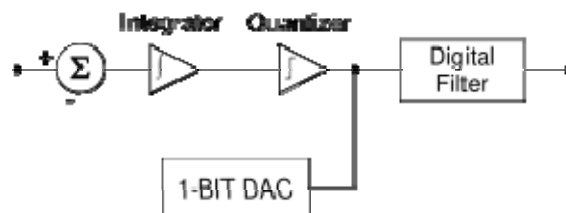


Bild 6.4.21 Delta-Sigma-Wandler

digitale Daten

Die digitalen Daten sind Ausgang jeder Analog-Digital-Wandlers.



Bild 6.4.22 Beispiel der digitalen Daten

dynamischer Fehler

Die dynamischen Fehler sind die Fehler, die sich zufällig ändern. Wir können sie nicht entfernen.

Zwischen dynamische Fehler gehört z.B.: die Nachwirkung und die Schaltzeit,

Flip-Flop-Schaltung

Die Flip-Flop-Schaltung hat eine wichtige Beschaffenheit, den Speicher.

S	R	Q	der Zustand
0	0	q	der Speicher
0	1	0	die Nullung
1	0	1	die Stelliny
1	1	0	verboten Zustandt

Tabelle 6.4.3 Wertetafel der Flip-Flop-Shaltung RS

J	K	Q	der Zustand
0	0	q	der Speicher
0	1	0	die Nullung
1	0	1	die Stelliny
1	1	$\overline{q}$	

Tabelle 6.4.4 Wertetafel der Flip-Flop-Shaltung JK

C	D	Q
0	0	q
0	1	q
1	0	0
1	1	1

Tabelle 6.4.5 Wertetafel der Flip-Flop-Shaltung D

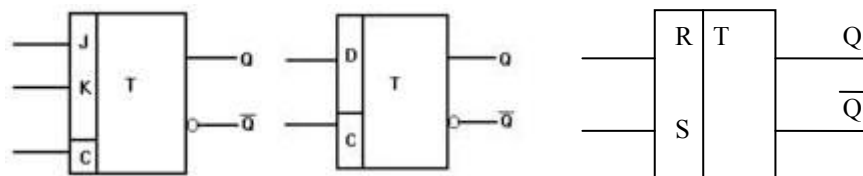


Bild 6.4.23 Flip-Flop-Schaltung

Gray-Code

Der Gray-Code ist Code, wo nur ein Wert ändern kann. Damit nicht der Fehler beim Lesen entsteht.

dekadische Wert	0	1	2	3	4	5	6	7
Gray-Code	0000	0001	0011	0010	0110	0111	0101	0100

Tabelle 6.4.6 Gray-Code

höchstwertiges Bit (MSB)

Es ist Bit mit größtem Wert. Beim Wandler ist MSB höchster Wert der Spannung.

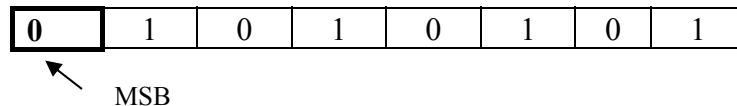


Tabelle 6.4.7 höchstwertiges Bit

Integrator

Der Integrator (der Tiefpassfilter) kann man als einen Sägezahngenerator benutzen. Die Integrators zählen den Mittelwert einiger Größe.

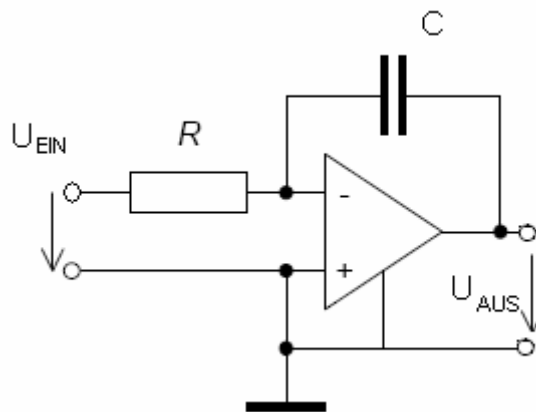


Bild 6.4.24 Integrator

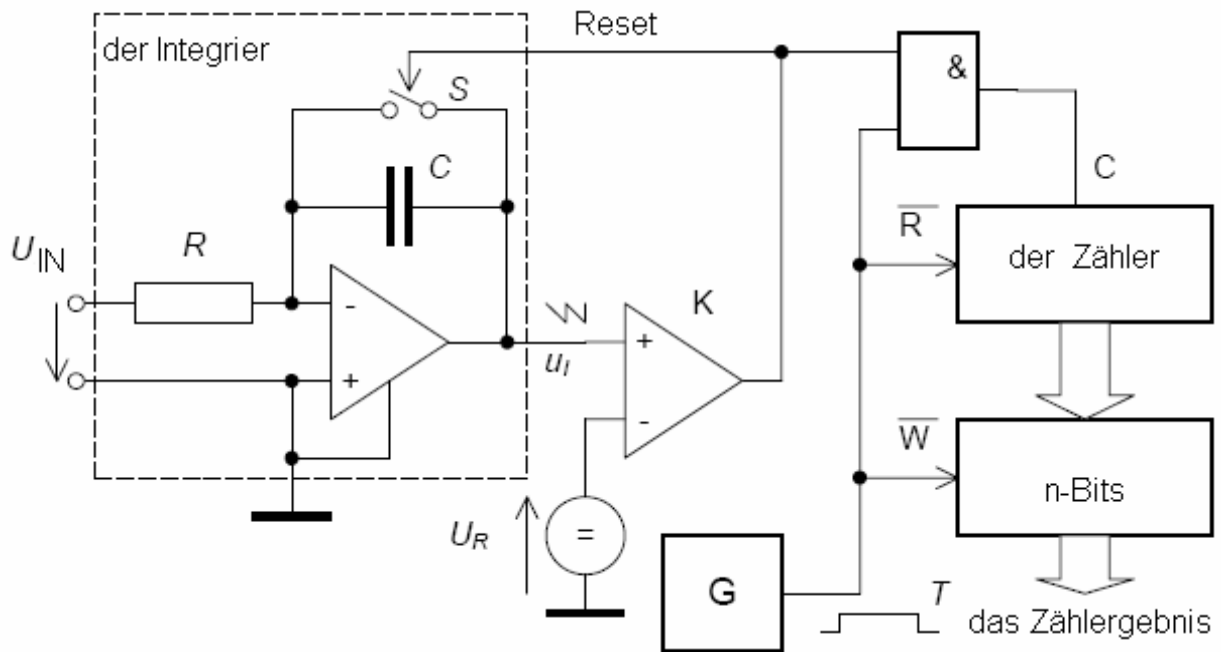
Integrierkomparator

Wenn wir die Spannung U_{IN} auf dem Eingang bringen, beginnt auf dem Ausgang des Integrier die Spannung erst auf den Wert $-U_R$ fallen. Der Komparator kippt um und schaltet den Schalter S. Der ganze Integrier gibt frei. Der Zähler zählt die Zyklen der Überschaltungen.

Am Ende schickt der Überschaltung die Daten auf den Ausgang. Die Frequenz des Integrierkomparator ist in der Form:

$$f = \frac{U_{IN}}{RC \cdot U_R} \quad (6.4.5)$$

wo: fdie Frequenz des Integriers
 U_{IN} ...die Spannung auf dem Eingang
 RCdie Größe der Elementen des Integriers
 U_Rdie Referenzspannung



Bildchen n. 6.4.25 Integrierkomparator mit der Übertragung auf die frequenz

Jitter T_S

Aus dem Bildchen 6.4.26 ist sichtbar, welchen Einfluss die aktive Zeit des Taktimpuls hat. Wenn die Zeit lange ist, kann man die Größe der Spannung ändern und so einen Fehler machen.

Der Dirac-Impuls hat keinen Jitter und dann ist für Abtastung der Beste.

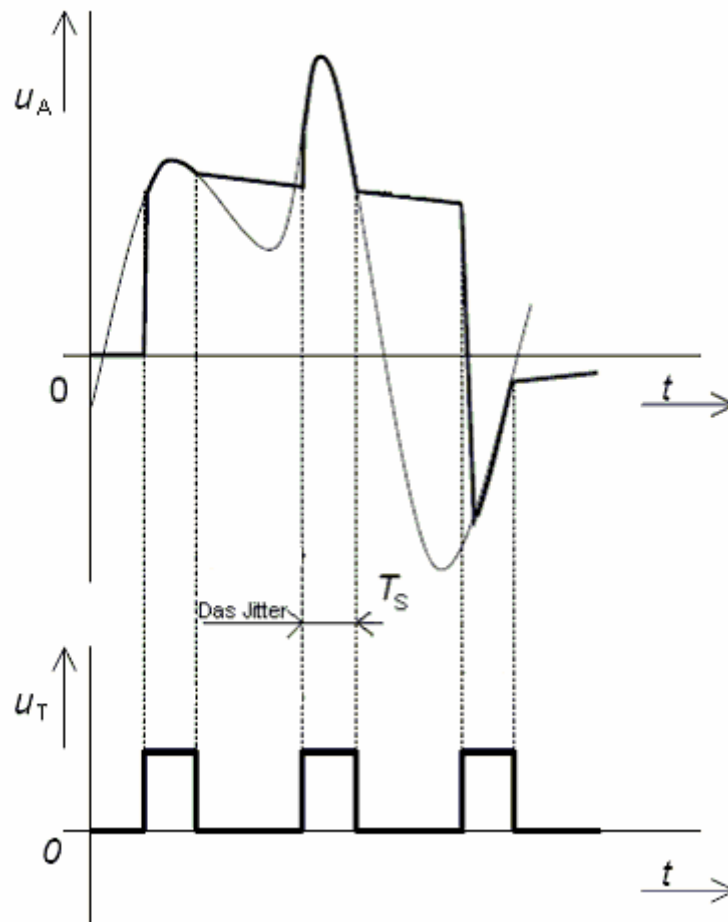


Bild 6.4.26 Jitter

Komparatorprinzip

Die Referenzspannung ist äquidistant verteilt zwischen den Widerständen auf die Quantisierungskennlinien. Nach der Zuführung der Spannung U_{IN} gliedert sich zwischen den Widerständen. Gemäß des Wertes der Spannung klappen die Komparators und schicken das Wort in den Koder.

Dieser Prinzip ist der schnellste Prinzip der Überschaltung auf das Digitalsignal. Komparatorprinzip ist teuer und anspruchsvoll auf der Zahl des Elementes.

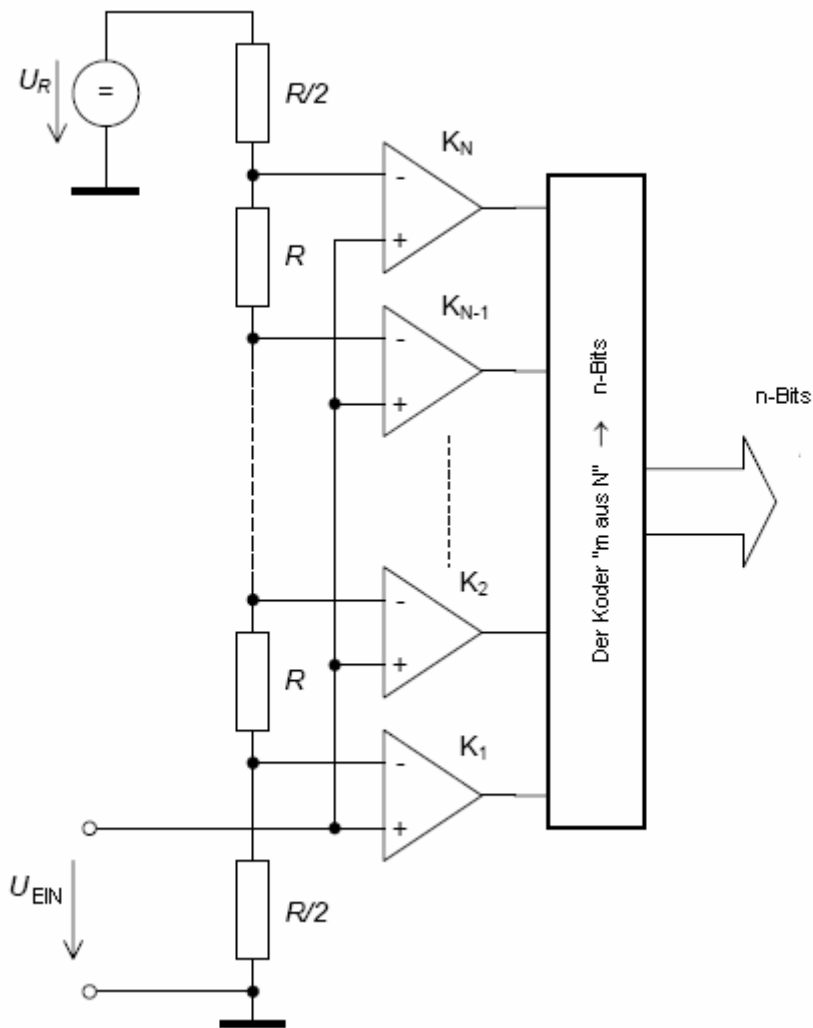


Bild 6.4.27 Komparatorprinzip des Analog-Digital-Wandlers

Kompensationprinzip

Der Kompensationprinzip beschreiben wir für 4-Bits Wandler.

Auf dem Beginn der Überschaltung wird der Wert des Wandlers auf 1000 angesetzt. Dann vergleicht man diesen Wert mit der Spannung auf dem Eingang. Wenn die Spannung grössere ist, schreibt man der Wert auf 1100 um und der ganze Zyklus kehrt erst in den Wert des LSB wieder. Wenn die Spannung kleiner ist, schreibt man der Wert auf 0100 um und der ganze Zyklus kehrt erst in den Wert des LSB wieder.

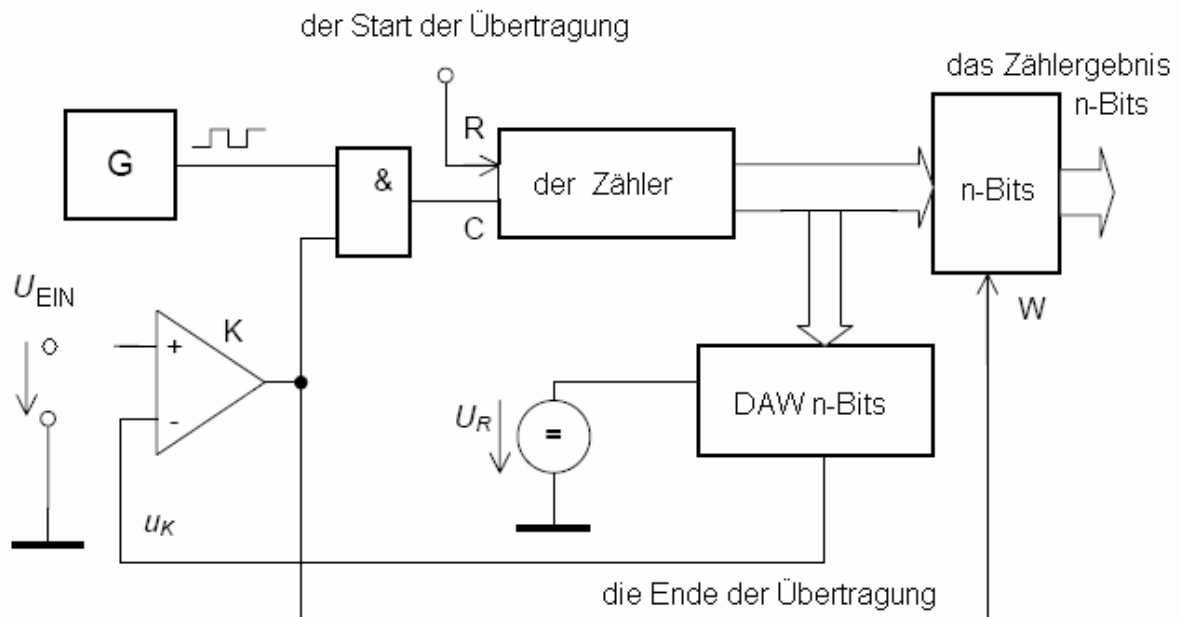


Bild 6.4.28 Kompensationsprinzip des Analog-Digital-Wandlers

Konvertierungsgeschwindigkeit

Es ist die Geschwindigkeit, hinter der Digitale Daten auf dem Ausgang nach dem Beginn der Übertragung erscheinen.

Die Konvertierungsgeschwindigkeit hängt von der Sorte der Übertragung ab. Der Parallelumsetzer ist schneller als Serialumsetzer.

Die Konvertierungsgeschwindigkeit hängt auch von den Übergangszustand des Elementes ab. Die Konvertierungsgeschwindigkeit gibt die Frequenz an, die der Wandler bearbeiten kann.

Näherungsprinzip

Dieser Prinzip vergleicht die Spannung U_{IN} mit der rückgekoppelten Spannung U_K so lange, bis es ähnlich ist.

Nach der Sorte teilen wir diese Wandler auf: die Zählschaltung, die Überwachungsschaltung und den Wandler mit der schrittweise Annäherung.

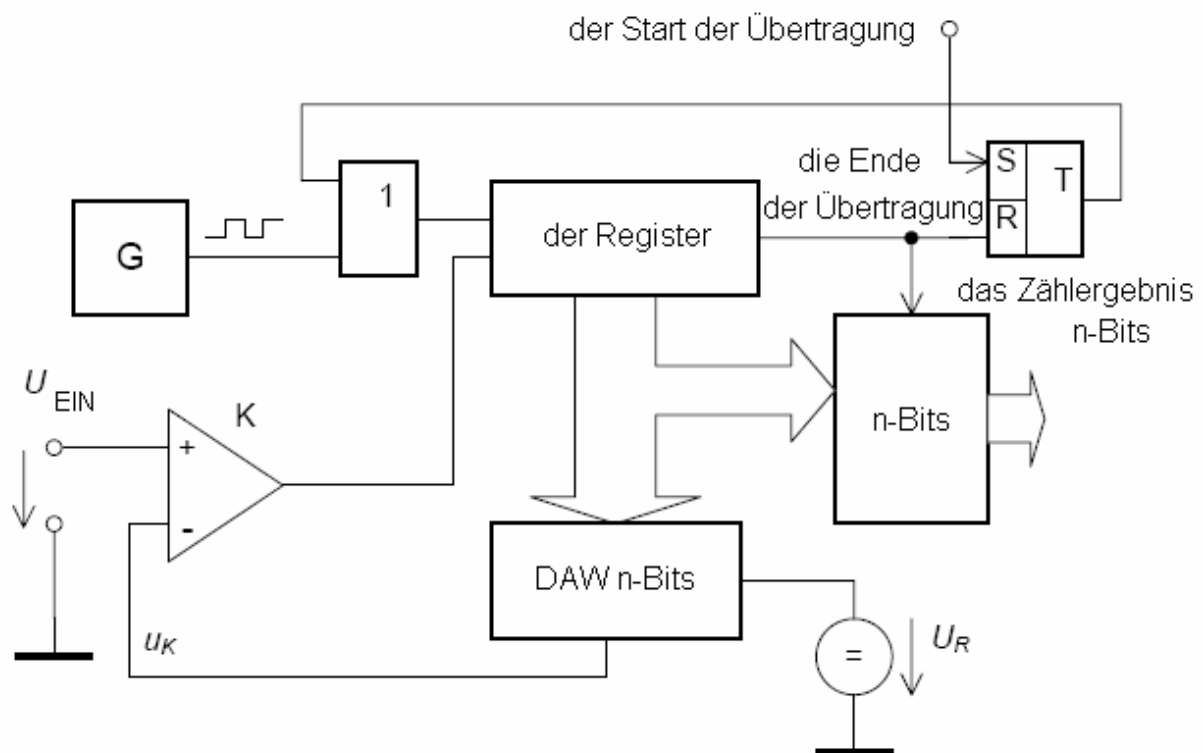


Bild 6.4.29 Näherungsprinzip des Analog-Digital-Wandlers

niedrigstwertiges Bit (LSB)

Es ist Bit mit dem kleinsten Wert. Beim Wandler ist LSB der niederste Wert der Spannung.

0	1	0	1	0	1	0	1
---	---	---	---	---	---	---	---

LSB
↖

Tabelle 6.4.8 niedrigstwertiges Bit

numerischer Code

Jeder Wandler überträgt seine Daten in den numerischen Code. Einige Codes haben die Fähigkeit der Feststellungen des Fehlers bei der Übertragung und können es reparieren.

Code
BCD-Code
Gray-Code
binärer Code

Tabelle 6.4.9 Beispiele der numerischen Codes

Parallelumsetzer

Der Parallelumsetzer ist schneller als Serialumsetzer, weil er alle Bits auf einmal bearbeitet. Dieser Umsetzer ist teurer und er hat viel Elementes.

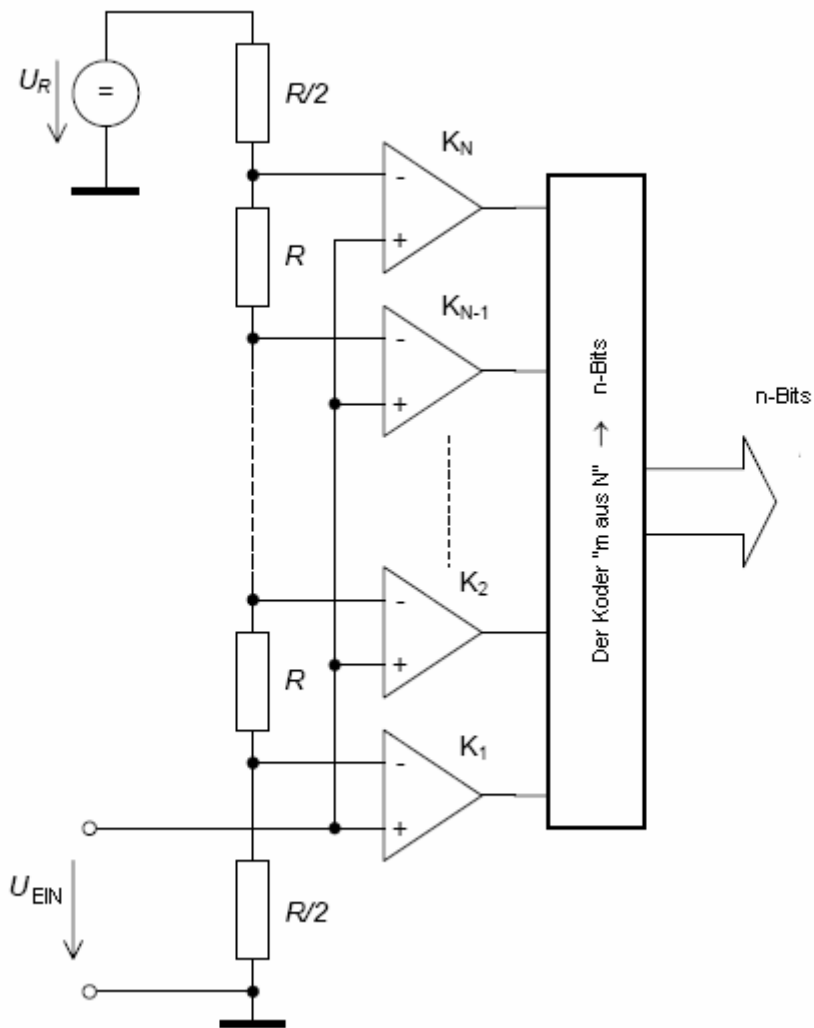


Bild 6.4.30 Parallelumsetzer

Quarzoszillator

Der Quarzoszillator können wir als Taktimpulsgenerator benutzen. Der Quarzoszillator ist genauer und stabiler als RC-Oszillator. Er kann höhere Frequenzen erzeugen.



Bild 6.4.31 Zeiche der Oszillator

Referenzgröße

Mehrheit der Komparators und des Wandlers gebrauchen für die Vergleichen der Referenzgröße der Spannung. Es kann man z.B. der Wert der Quantisierungskennlinie sein.

Register

Der Register ist das Element, das sich die Daten erinnert.

Der Register häuft einige Daten. Wenn die Umsetzungsvorgang ganz ist, schickt sie der Register zu nächster Bearbeitung.

Rückführschaltung

Die Rückführschaltung bringt den Wert auf dem Ausgang zurück auf den Eingang. Die Rückführschaltung kann Eigenschaften der Schaltung aufbessern oder sie kann die steuerndene Information für den Eingang bringen.

Sägezahngenerator

Der Sägezahngenerator ist ein Generatog, der Ausgangsignal in der Form Sägezahn macht. Dieses Signal kann man sein für den Wandler mit der Zweitaktintegration benutzt werden.

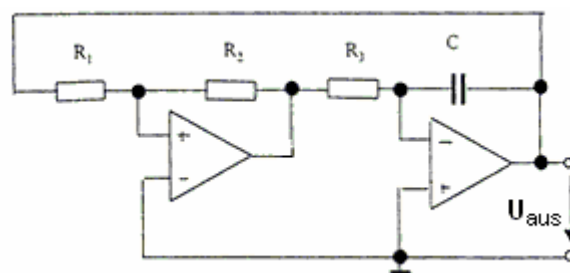


Bild 6.4.32 Einfacher Sägezahngenerator

Schaltlaufwerk

Das Schaltlaufwerk steuert ganze Schaltung. Es synchronisiert einzelne Blöcke. Es kommuniziert mit der Umgebung und stellt richtige Übersetzung sicher.

der Schaltungsaufwand

Wir haben zwei Variante: die Bipolartechnik
 die Unipolartechnik

Die Bipolartechnik ist teuer, sie braucht viele Energie und sie ist schneller als die Unipolartechnik. Die Unipolartechnik ist billiger, sie braucht weniger Energie, aber sie ist langsamer als die Bipolartechnik.

Taktimpuls

Mit Hilfe des Taktimpuls synchronisiert das Schaltlaufwerk den Wandler.

Die Taktimpulse sind mit Hilfe der Quartzzillator generiert.

Wir fordern höhe Steilheit, stabile Frequenz und gleich bleibende Periode T_V , bleibende Impulsverhältnis und bleibenden Puls T_S .

Impulsverhältnis ist Verhältnis zwischen der Zeit des Puls T_S und der Zeit der Periode T_V

$$n = \frac{T_S}{T_V} \cdot 100 \quad (6.4.6)$$

wo: nImpulsverhältnis

T_S ...Zeit des Puls
 T_V ...Zeit der Periode

Jede Schaltung reagieren auf verschiedene Änderungen. Manchmal reagieren sie auf die Impulsvorderflanke, manchmal auf abfallende Flanke, manchmal auf ganzen Puls.

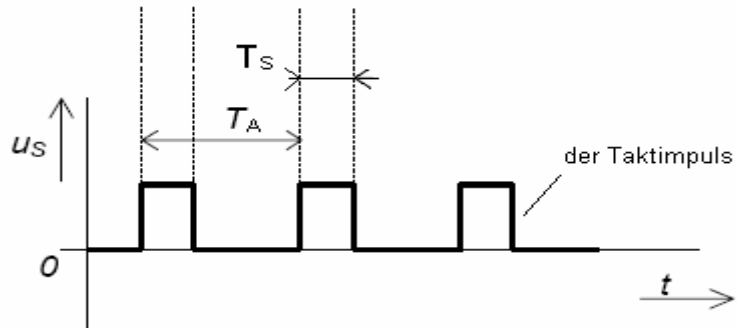


Bild 6.4..3 Taktpuls

temperaturabhängig

Diese Abhängigkeit gehört zwischen dynamische Fehler, genauer zwischen die Fehler der Verstärkung. Wir sterben diese Abhängigkeit minimisieren.

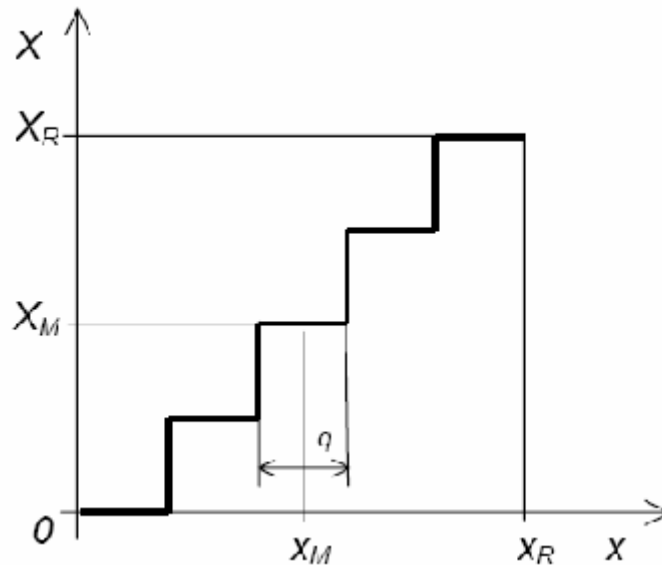
Diese Abhängigkeit hat nicht z.B. der Wandler mit der Zweitaktintegration.

Übersetzungsfehler

Übertragung des Analogsignals auf den Digitalsignal ist eigentlich die Abrundung einzelner Werten. Im idealem Zufall ist der Übersetzungsfehler der Wert des 0,5 LSB.

Umrechnungscharakteristik

Die Umrechnungscharakteristik ist die Staffelfunktion. Die Distanz der Quantisierungskennlinien heißt der Quantisierungsschritt q . Er entspricht dem Wert des LSB. Bei der Quantisierung runden wir analogischen Wert x_M auf den digitalischen Wert X_M ab.



- wo: x_M ... Augenblickswert der Eingangsspannung
 x_R ... maximale Eingangsspannung
 X_M ... Größe der Ausgangsspannung des Wandler
 X_R ... maximale Größe der Ausgangsspannung des Wandler
 q ... Quantisierungsgeräusch

Bild 6.4.34 Umrechnungscharakteristik

Umsetzungsgeschwindigkeit

Die Umsetzungsgeschwindigkeit gibt an, wieviel Daten (im Bits) man in einer Sekunde überträgt.

$$v_u = \frac{N}{t_u} \quad (6.4.7)$$

- wo: v_uUmsetzungsgeschwindigkeit
 NZahl des übertragenen Bits
 t_nZeit der Übertragung.

Umsetzungsvorgang

Der Umsetzungsvorgang ist das Ergebnis der Übertragung. Es ist ein Wort, das die nütze Daten, die Steuerdaten und z.B. Daten für die Ermittlung des Fehler beinhaltet.

Die Ermittlung des Fehlers kann z.B. Paritätsbit machen. Es existiert geradzahlige Parität und ungeradzahlige Parität.

Wir haben z.B. geradzahlige Parität. Wenn die Summe des Bits gradzahlig ist, stellt sich das Paritätsbit auf 1 ein. Wenn die Summe des Bits ungeradzahlig ist, stellt sich das Paritätsbit auf 0 ein. Für das ungeradzahlige Parität ist es umgekehrt.

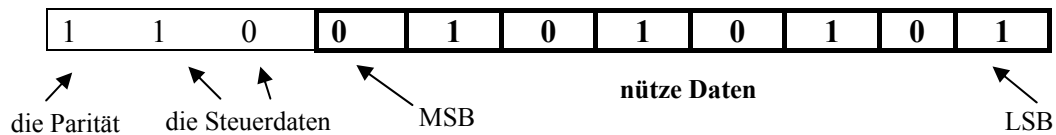


Bild 6.4.35 Umsetzungsvorgang

Umsetzungszeit

Die Umsetzungszeit ist die Zeit, in der der Wandler befähigt ist einen Abtast oder einen Umsetzungsvorgang zu produzieren.

Unipolartechnik

Die Bipolartechnik gibt unipolare Elemente an.

Die unipolartechnik ist billiger, sie braucht wenige Energie und sie ist langsamer als die bipolarstechnik. Versorgungsspannung des Komparators kann nicht negativ sein und dann kann nicht negatives Analogsignal verarbeiten.

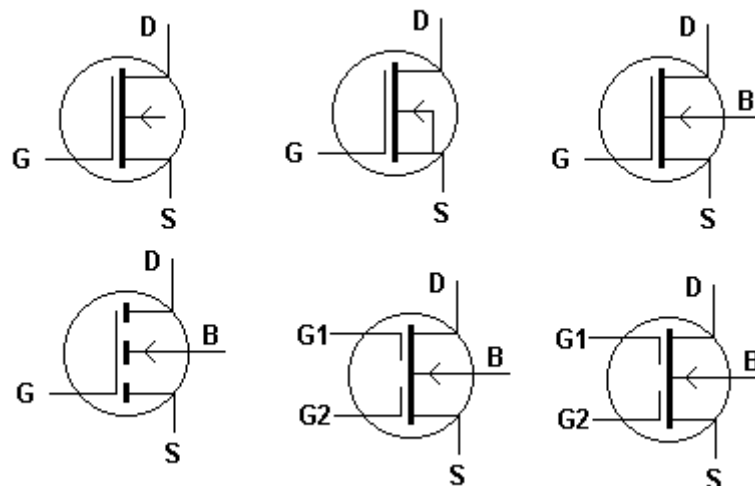


Bild 6.4.36 Beispiele der Unipolartransistors

Unterscheidungsvermögen

Die Unterscheidungsvermögen ist der kleinster Wert, den der Wandler noch unterscheiden kann.

Die Unterscheidungsvermögen bei dem Wandler ist der Wert des LSB.

Zähler

Der Zähler rechnet die Impulse eines Zyklus. Wenn der Zyklus zu Ende ist, überträgt er sein Ergebnis in binären Code oder er gibt seine Angabe über die Zahl der Pulses nächsten Elemente über.

Zählergebnis

Das Zählergebnis oder der Umsetzungsvorgang ist das Ergebnis der Übertragung. Es ist ein Wort, das die nütze Daten, die Steuerdaten (Startbit, Synchronisierbit) und z.B.

Daten für die Ermittlung des Fehlers beinhaltet.



Bild 6.4.37 Zählergebnis

Zählverfahren Wandler

Dieser Wandler ist ganz schnell und hat weniger Elemente als Paraelumsetzer. Seiner Prinzip kann man z.B. für 8 Bits Wandler erklären. ADW 1 macht die Übertragung höherer 4 Bits der Eingangsspannung. Der Ausgang des ADW1 ist rekonstruiert mit Hilfe der DAW. Rekonstruierte Spannung rechnet von Eingangsspannung ab. Der Rest der Spannung verstärkt sich (im unserem Zufall 16 mal) und der ADW überträgt unterer 4 Bits. Beide Hälften stellen sich zusammen und wir haben den Umsetzungsvorgang über 8 Bits.

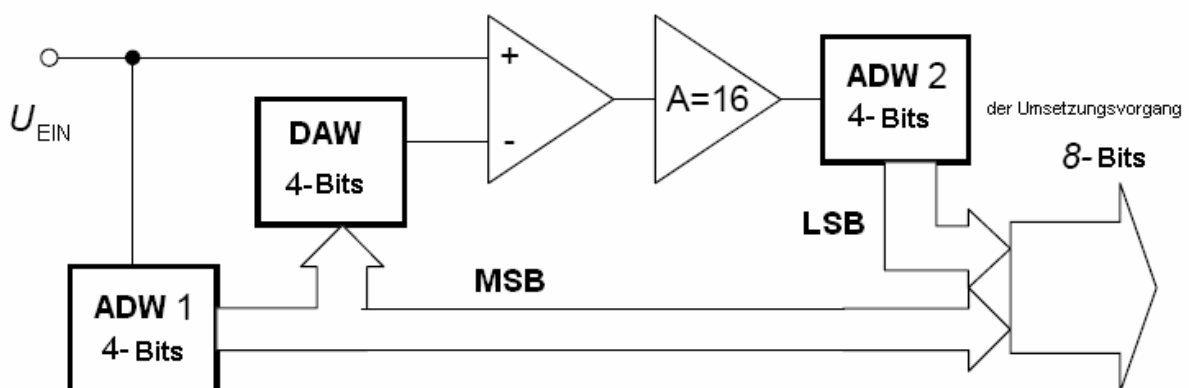


Bild 6.4.38 Zählverfahren Wandler

Zweitaktintegration

Dieser Wandler arbeitet im zwei Takten.

Im ersten Takt ist auf den Eingang des Integrators die Eingangsspannung U_{IN} herbeigeführt. Der Zähler zählt die Impulse aus dem Quarzoszillator G. Die Spannung u_i wächst und dieser Vorgang dauert bis in den Überlauf des Komparators K. Die Zeit T_1 ist also solide unter der Formel (6.4.8) festgelegt.

$$T_1 = \frac{2^n}{f_G} \quad (6.4.8)$$

wo: T_1Zeit erstes Taktes
 nZahl des Bits des Zählers
 f_GFrequenz der Quarzoszillator

Im zweiten Takt ist auf dem Eingang des Integriers die Referenzspannung U_R mit umgekehrter Polarität als Eingangsspannung U_{IN} herbeigeführt. Der Zähler zählt immer die Impulse. Die Spannung u_i fällt erst zum Nullwert. Der Zweite Takt ist zu Ende und auch ganze Übertragung. Die Zeit T_2 des zweiten Takts gibt Formel (6.4.9) an.

$$T_2 = \frac{N}{f_G} = \frac{U_{IN}}{U_R} \cdot T_1 \quad (6.4.9)$$

wo: T_2Zeit zweites Taktes
 NZahl des einlesungen Bits des Zählers
 f_GFrequenz der Quarzoszillators
 U_{IN}Eingangsspannung
 U_RReferenzspannung
 T_1Zeit erstes Taktes

Aus der Formel (6.4.10) bekommen wir die Zahl des einlesungen Bits des Zählers.

$$N = T_2 \cdot f_G = N_Z \cdot \frac{U_{IN}}{U_R} \quad (6.4.10)$$

wo: NZahl des einlesungen Bits des Zählers
 T_2Zeit zweites Taktes
 f_GFrequenz des Quarzoszillators
 U_{IN}Eingangsspannung
 U_RReferenzspannung
 N_ZKapazität des Zählers

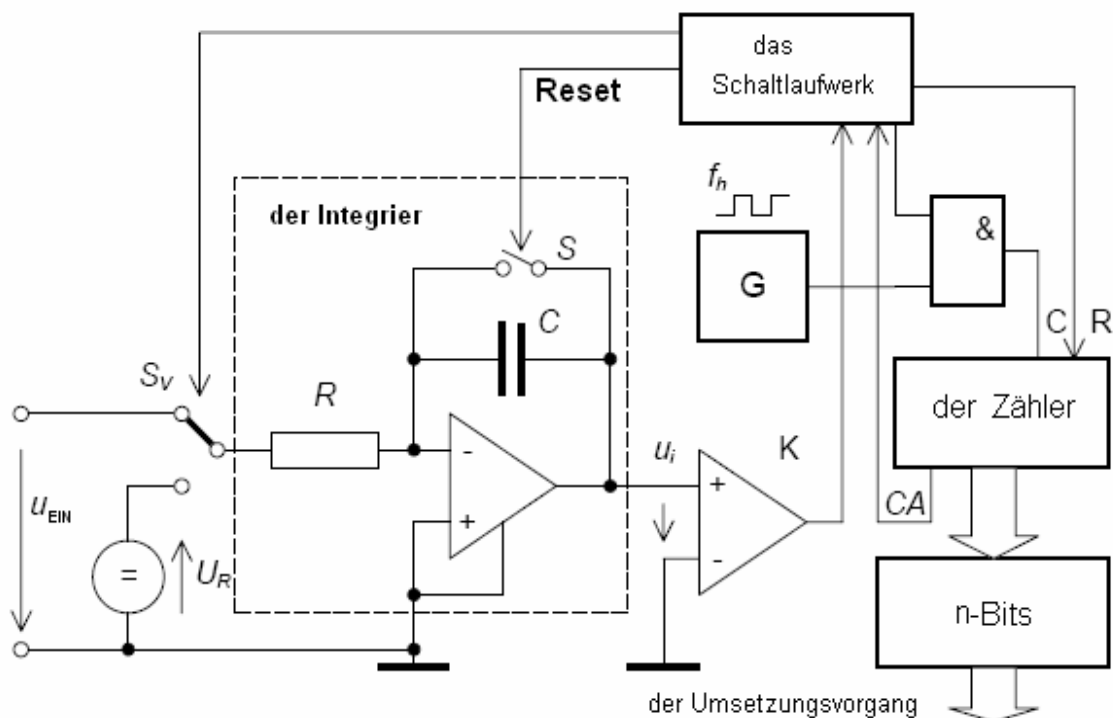


Bild 6.4.39 Wandler mit der Zweitaktintegration

Dieser Wandler eliminiert die Zeitunsicherheit der Elementes R und C und eliminiert Instabilität der Frequenz des Quarzoszillators. Die Größe des Umsetzungsvorgangs ist nur auf der Kapazität des Zählers, der Größe der Eingangsspannung und der Größe der Referenzspannung abhängig.

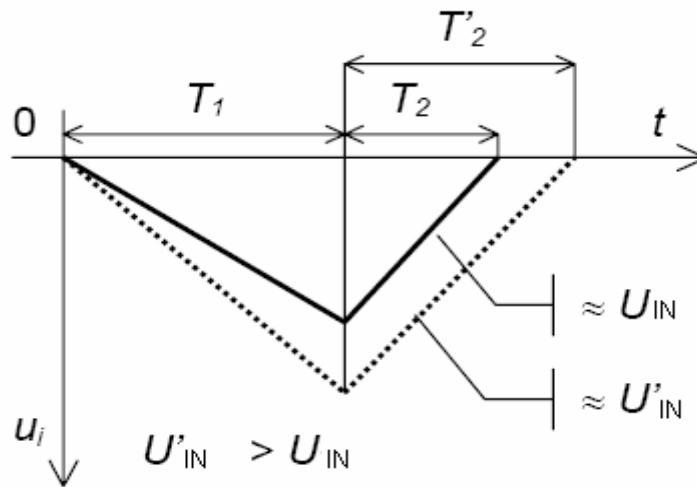


Bild 6.4.40 Verlauf der Spannung auf dem Integrierer

6.5 Digital-Analog-Wandler

alternative Linearcharakteristik

Bei der Bestimmung des Fehlers des Analog-Digital-Wandlers benutzen wir die alternative Linearcharakteristik.

Es ist die ideale Charakteristik des Analog-Digital-Wandlers die über den Nullpunktfehler verschoben wird. Die alternative Linearcharakteristik hat meistens eine andere Neigung, als ideale Charakteristik.

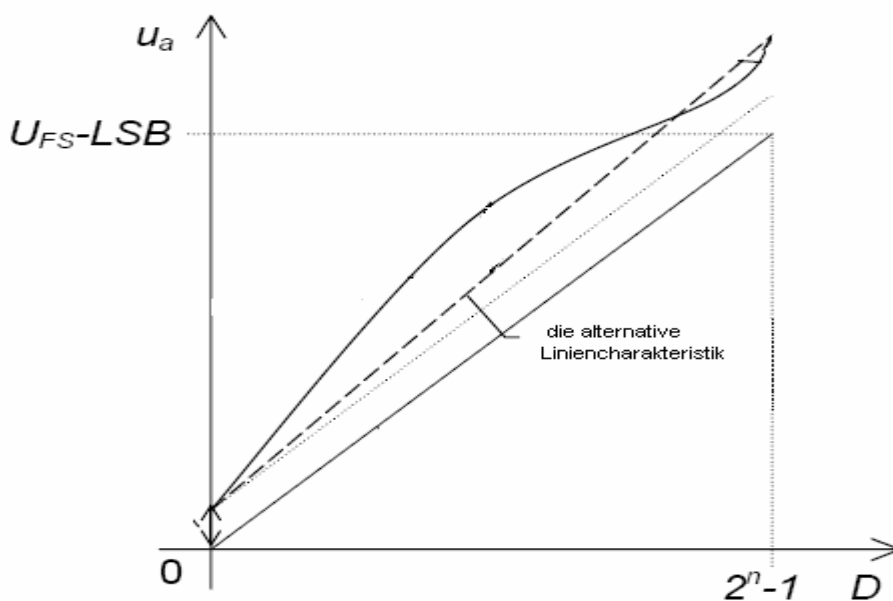


Bild 6.5.1 alternative Linearkarakteristik des Digital-Analog-Wandlers

analoges Ausgangssignal

Das analoge Ausgangssignal ist ein Produkt der Übertragung des Digital-Analog-Wandlers. Als das Analogsignal durch die Kantenglättung durchgeht, hat das Signal die Form der Staffelfunktion. Die Kantenglättung dieser Stufen glättet aus und auf dem Ausgang erscheint fast vollkommen analoges Ausgangssignal.

Auflösungsvermögen

Die Auflösungsvermögen ist minimaler Wert, den Wandler auflösen kann. Bei den Wandlern ist es der Wert des LSB.

Decoder

Der Decoder ist eine Schaltung, die einigen binären Code lösen kann.

Digital-Analog-Wandler

Der Digital-Analog-Wandler ist eine Schaltung, die das digitale Eingangssignal auf das analoge Ausgangssignal übertragen kann.

Ganzes Digital-Analog-Wandler hat drei Teile. Der Digital-Analog-Wandler, der allein Übertragung macht. Der Tiefpassfilter (oder Kantenglättung), der die Staffelfunktion glättet, die aus dem Wandler geht. Letztes Element ist das Schaltlaufwerk, das ganz Wandler leitet.

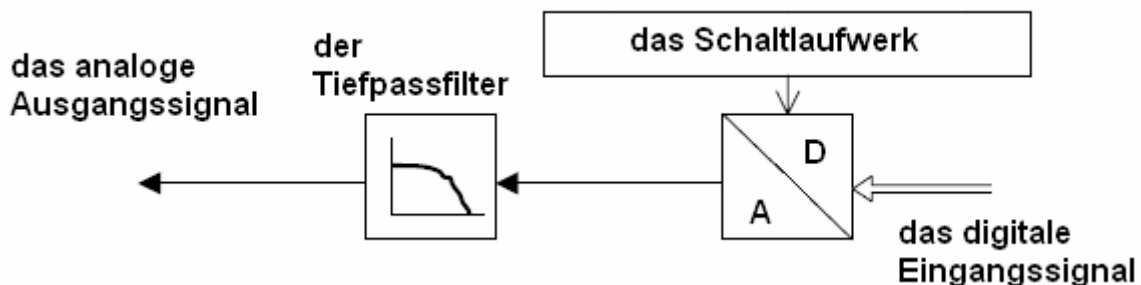


Bild 6.5.2 Digital-Analog-Wandler

digitales Eingangssignal

Auf den Eingang des Wandlers kommt das digitale Eingangssignal.

Es ist ein Wort, das die nützlichen Daten, die Steuerdaten (Startbit, Synchronisierbit) und z.B. Daten für die Ermittlung des Fehlers beinhaltet.

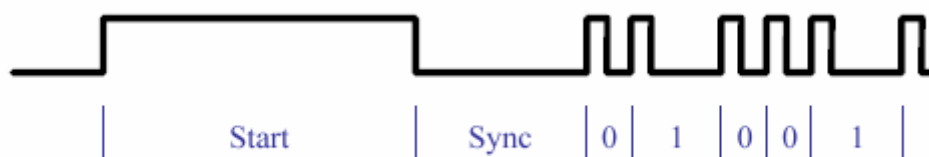


Bild 6.5.3 Digitales Eingangssignal

effektive Anzahl von Bits

Wenn die Eingangsgröße schnell ändert, können sich die dynamische Fehler hier durchsetzen. Das bedeutet, dass mit wachsenden Frequenzen die Auflösungsvermögen des Wandlers fällt. Deshalb leiten wir die effektive Anzahl von Bits in der Abhängigkeit von der Frequenz. Ein 8-Bit-Wandler kann z.B. auf der Frequenz im MHz nur eine effektive Anzahl von Bits 6 haben.

Fehler der Verstärkung

Bei der Bestimmung des Fehlers des Analog-Digital-Wandlers benutzen wir den Fehler der Verstärkung.

Es ist die Differenz zwischen den Neigungen der idealen Charakteristik und der alternativen Linearcharakteristik.

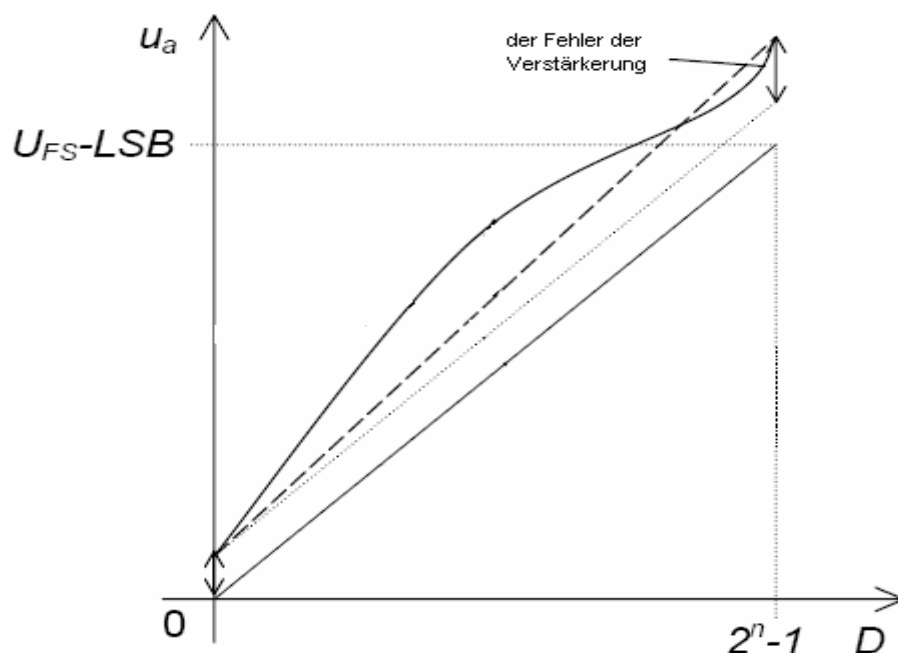


Bild 6.5.4 Fehler der Verstärkung des Digital-Analog-Wandlers

geketteten Stand

Die Digital-Analog-Wandler gebrauchen analogen Schalter für die Einschaltungen einzelner Spannungsbereichen. Die Zeite der Abschaltung und der Verspannung müssen zum Nullwert nahekommen. Der gekettete Stand muss möglichst wenigen Widerstände haben.

getrennten Stand

Die Digital-Analog-Wandler gebrauchen analogen Schalter für die Einschaltungen einzelner Spannungsbereichen. Die Zeite der Abschaltung und der Verspannung müssen zum Nullwert nahekommen. Der getrennte Stand muss größtmöglichen Widerstände haben.

ideale Charakteristik

Bei der Bestimmung des Fehlers des Analog-Digital-Wandlers benutzen wir die ideale Charakteristik.

Die ideale charakteristik ist die Charakteristik, die wir bekommen streben. In der Praxi ist es unmöglich.

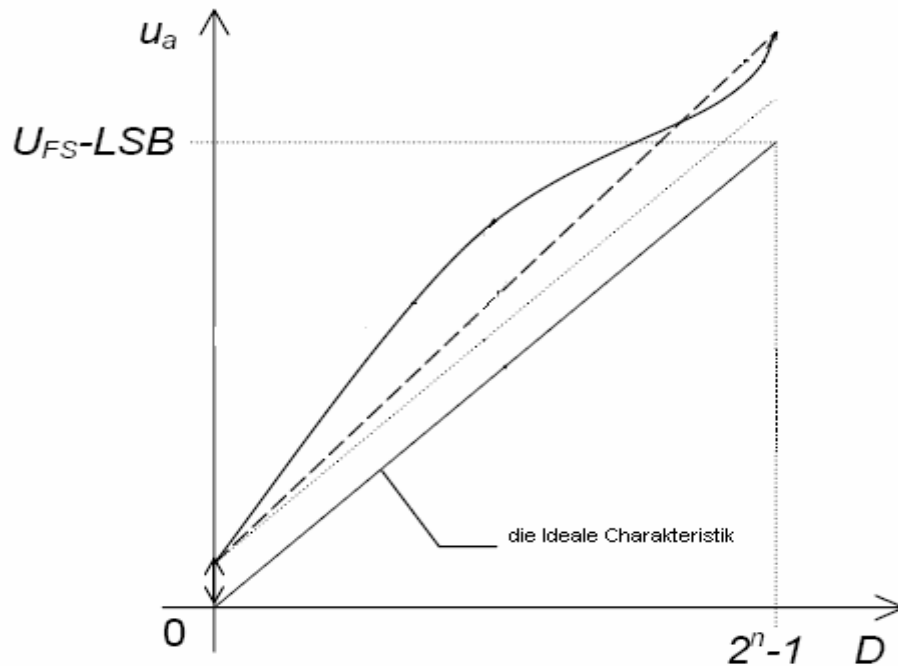


Bild 6.5.5 ideale Charakteristik des Digital-Analog-Wandlers

Nachwirkung

Die Nachwirkung gehört zu den statischen Fehler. Der Komparator geht aus dem geketteten Standt in den getrennte Standt bei anderem Wert der Spannung als bei dem Übergang aus dem getrenntet Standt auf den geketten Standt. Die Diferenz dieser Spannungs heißt die Nachwirkug.

Nichtlinearität

Bei der Bestimmung des Fehlers des Analog-Digital-Wandlers benutzen wir die Nichtlinearität.

Es ist wirkliche Gestalt der Übersetzungscharakteristik.

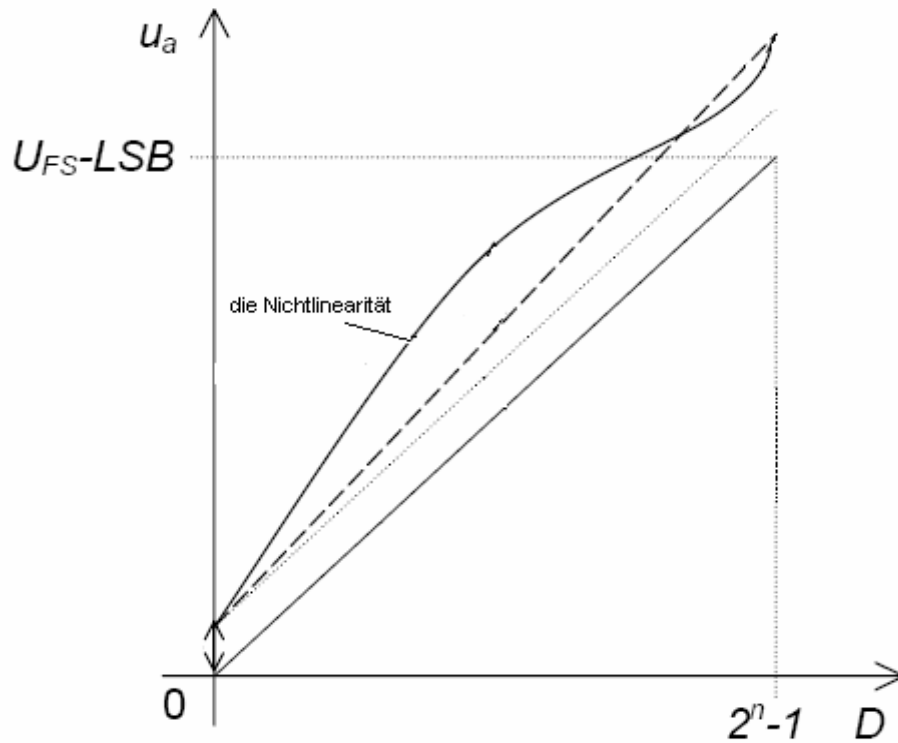


Bild 6.5.6 Nichtlinearität des Digital-Analog-Wandlers

Nichtlinearitätsfehler

Bei der Bestimmung des Fehlers des Analog-Digital-Wandlers benutzen wir den Nichtlinearitätsfehler.

Es ist die maximale Abweichung der wirklichen Charakteristik von der alternativen Linearcharakteristik.

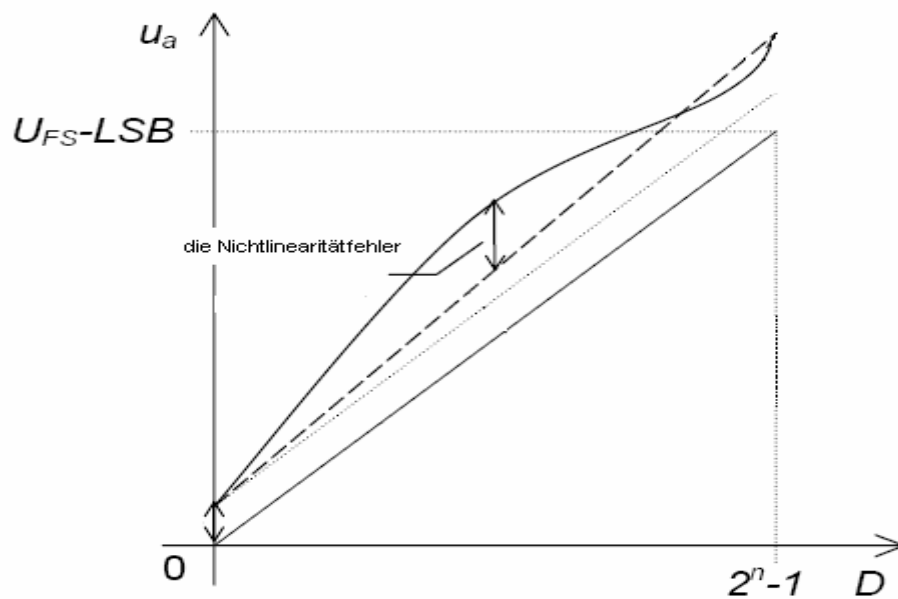


Bild 6.5.7 Nichtlinearitätsfehler des Digital-Analog-Wandlers

Nullpunktfehler

Bei der Bestimmung des Fehlers des Analog-Digital-Wandlers benutzen wir den Nullpunktfehler.

Es ist die Verschiebung der Übersetzungscharakteristik über den konstanten Wert. Bei heutigen Wandlern können wir diesen Fehler abschaffen.

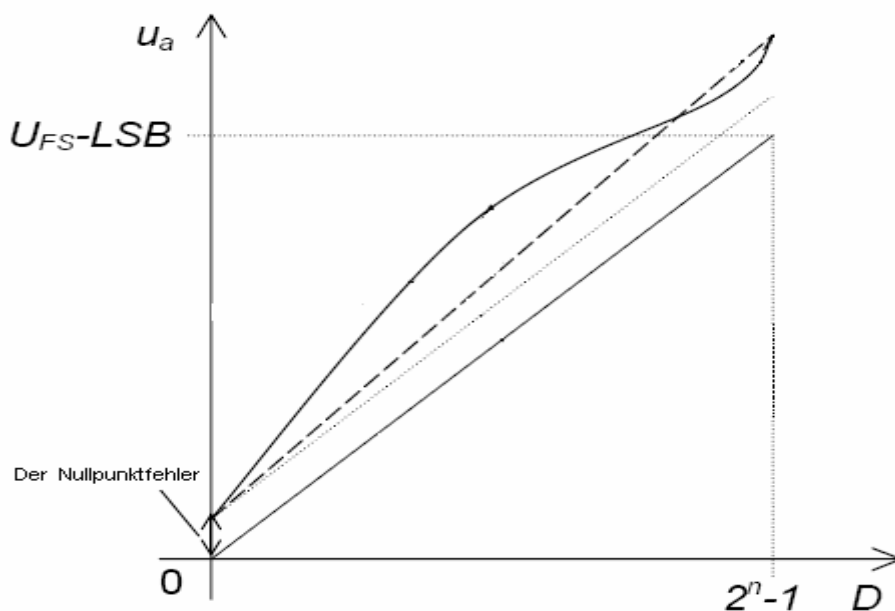


Bild 6.5.8 Nullpunktfehler des Digital-Analog-Wandlers

Mehrrampenverfahren

Diese Wandler nutzen die Referenzspannung und das Widerstandnetzwerk aus. Mit Hilfe dieser Widerstände haben wir binäre abgestufte Stromquellen. Diese Quellen schalten gemäß dem digitalen Eingangssignal auf den Ausgang des Wandlers. Hier werden sie zusammengezählt und auf die Spannung übertragen.

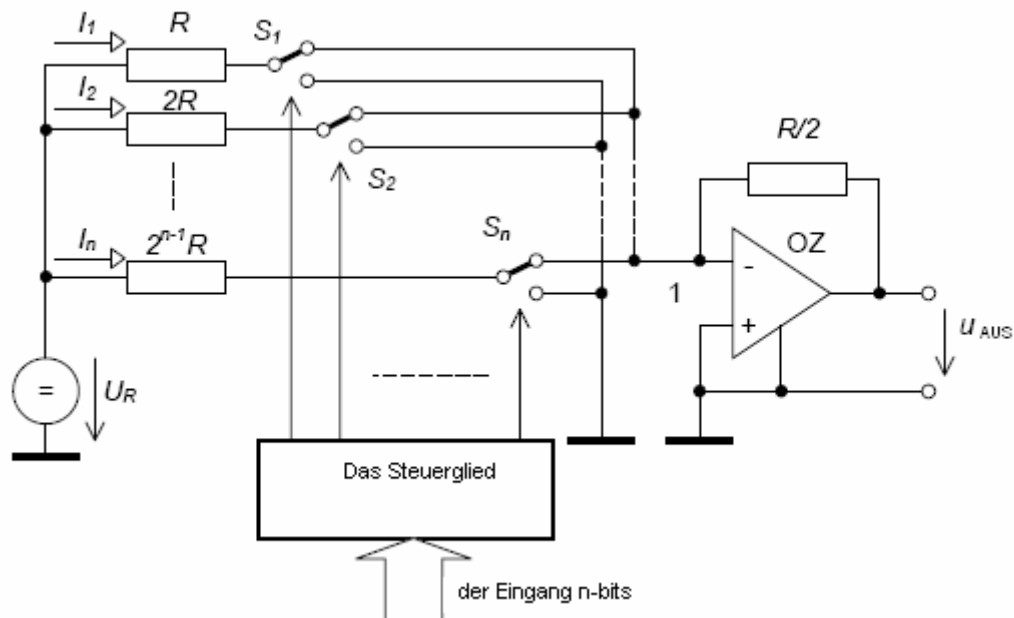


Bild 6.5.9 Mehrrampenverfahren

Operationsverstärker

Der Operationsverstärker ist ein Element, das verschiedene Verwendungen hat. Wir können ihn als z.B. den Komparator, den Verstärker, das Differenzglied und die Zählschaltung gebrauchen.

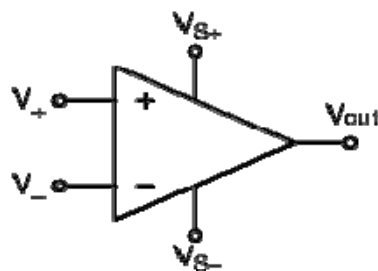


Bild 6.5.10 Operationsverstärker

Quelleinhaltung

Die Quelleinhaltung ist wichtig für die rechte Übertragung des digitalen Signals. Wir brauchen, damit die Quelle ausreichend hart war und damit seine Temperaturabhängigkeit klein war.

R2R-Netzwerk

Das R2R-Netzwerk kann die Eingangsspannung auf das Verhältnis 1:1 teilen.

Wenn wir Wandler mit dem Mehrverfahren mit dem R2R-Netzwerk gebrauchen, machen wir nicht den Fehler der Anwendungen der Widerstände mit verschiedenen Toleranzen. Das R2R-Netzwerk kann nur zwei Werte der Widerstände gebrauchen.

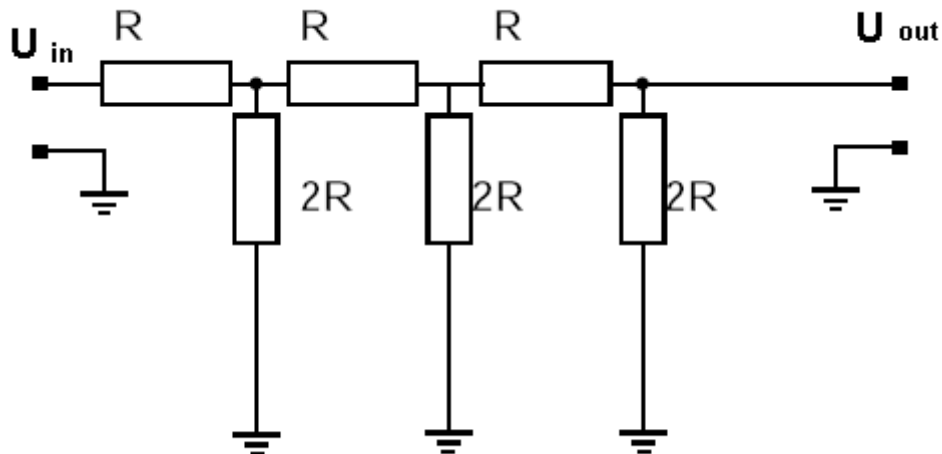


Bild 6.5.11 R2R-Netzwerk

statischer Fehler

Die statische Fehler sind die Fehler, die durch die Ungenauigkeiten des Elementes entstehen. Ihre Größe ändert sich nicht. Wir können sie teilweise entfernen.

Die statische Fehler sind: der Fehler der Verstärkung, die Nichtlinearität und der Offsetdruck.

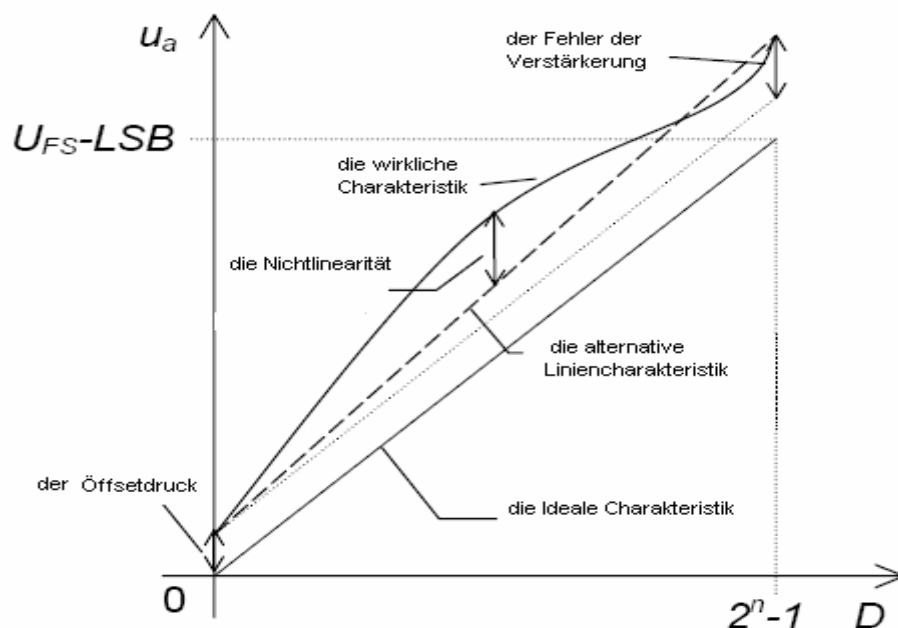


Bild 6.5.12 Statische Fehler des Digital-Analog -Wandlers

Steuerglied

Das Steuerglied ist eigentlich der Schalter, gemäß der Größe des digitalen Eingangssignals, die einzelne Äste des Wandlers schaltet.

Überschaltzeit

Die Überschaltzeit ist die Zeit, in die die Schaltung aus dem geketteten Standt in den getrennten Standt oder aus dem getrennteten Standt in den geketteten Standt übergeht.

Widerstandnetzwerk

Das Widerstandnetzwerk kann die Eingangsspannung auf die Verhältnisse $1:2:4:8:\dots:2^{n-1}$ teilen, was eigentlich binären Code angibt.

Wenn wir Wandler mit dem Mehrverfahren mit dem Widerstandnetzwerk gebrauchen, machen wir den Fehler der Anwendungen des Widerstände mit verschiedenen Toleranzen.

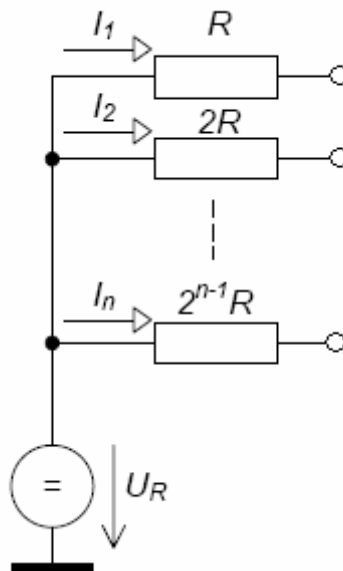


Bild 6.5.13 Das Widerstandnetzwerk

wirkliche Charakteristik

Bei der Bestimmung des Fehlers des Analog-Digital-Wandlers benutzen wir die wirkliche Charakteristik.

Die wirkliche Charakteristik ist wirkliche Form der Übersetzungscharakteristik gegebenes Wandlers. Wir beurteilen ihre Nichtlinearität zu der alternativen Linaercharakteristik.

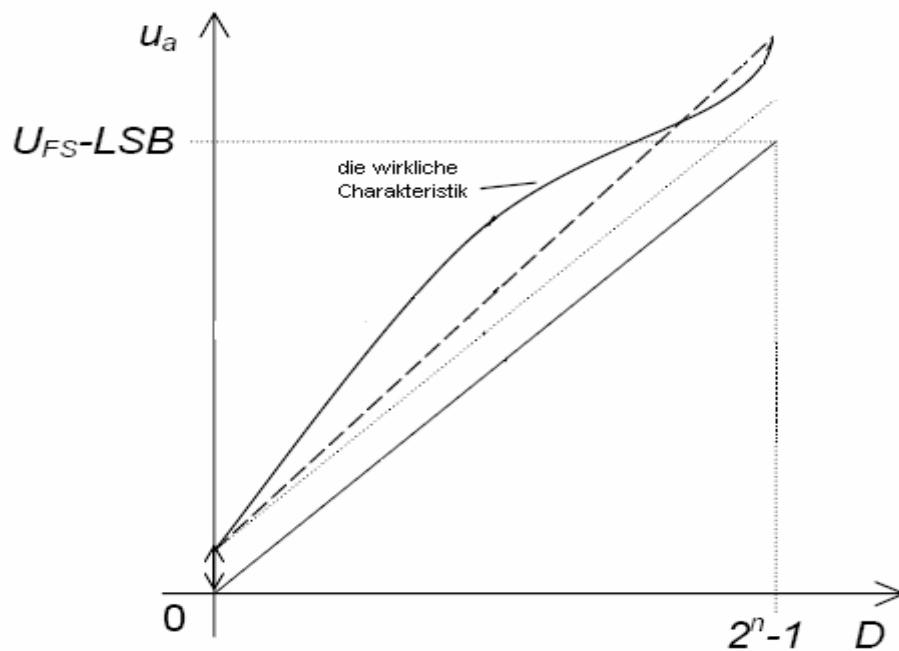


Bild 5.5.14 wirkliche Charakteristik des Digital-Analog-Wandlers

6.6 Spektralanalyse

Amplitudenspektrum

Das Amplitudenspektrum sagt, welche Größe eigentliche Oberwellen haben.

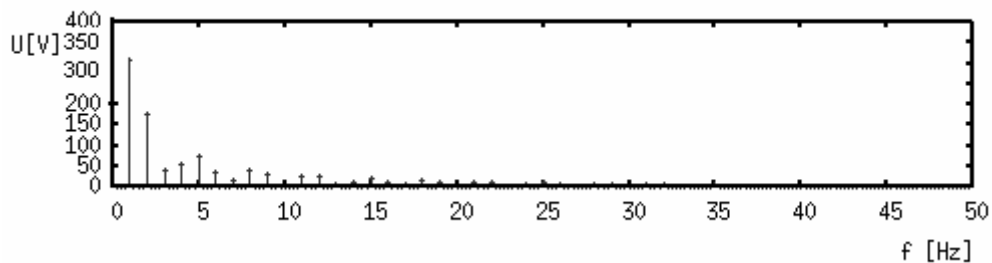


Bild 6.6.1 Das Amplitudenspektrum

Oberwelle

Jedes unharmonische Signal können wir auf die Oberwellen gliedern. Jede Oberwelle hat ihre Frequenz ihre Amplitude.

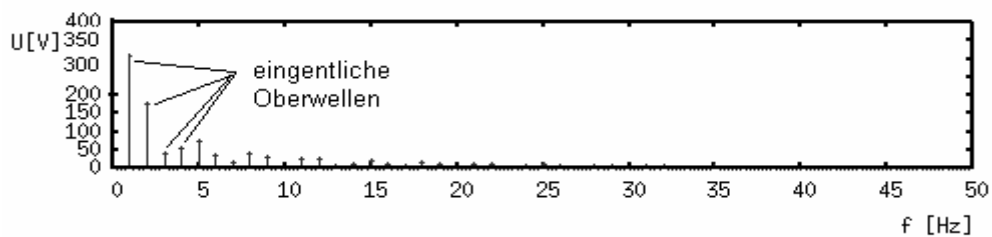


Bild 6.6.2 Oberwellen

Spektralanalysator

Der Spektralanalysator ist eine Vorrichtung, die die Spektralanalyse macht. Der Spektralanalysator kann das Spektrum im Frequenzgebiet darstellen.

Spektralanalyse

Die Spektralanalyse dient zur Ermittlung der Bedeutung eigentlichen Oberwellen des Signals. Wir unterscheiden das Amplitudenspektrum und Frequenzspektrum. Jedes harmonisches Signal ist mit Hilfe der zweite komplex konjugierte Zahlen ausgedrückt.

$$s(t) = \frac{1}{2} C_1 \cdot e^{j\omega t} + \frac{1}{2} C_1^* \cdot e^{-j\omega t} \quad (6.6.1)$$

wo: $s(t)$...harmonisches Signal im Frequenzgebiet
 C_1Amplitude des Signals
 ωWinkligeschwindigkeit
 tZeit

Spektrallinie

Mit Hilfe der Spektrallinie darstellen wir eigentliche Oberwellen bei der Spektralanalyse. Die Länge der Spektrallinie gibt die Größe gegebenes Elementes an.

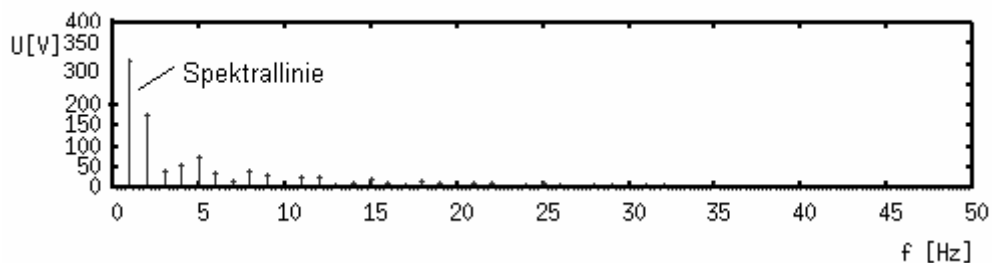


Bild 6.6.3 Spektrallinie

Spektrum

Das Spektrum des Signals gibt den Stellvertretung eigentlicher Oberwellen des Signals im Frequenzgebiet an.

7 Rozvaha nad softwarovým řešením slovníku

V rámci BB2E jsem měl také posoudit, jakou formou budu mou práci prezentovat v multimediální podobě. Slovník jsem zkoušel vytvořit v HTML editoru, PHP editoru a C++ Builderu.

HTML editor je asi nejjednodušší forma, ve které je možno program realizovat. Program by určitě splnil nejjednodušší funkce, jako je základní prohlíže při klikání na jednotlivé odkazy. Samotné programování je také velice jednoduché a nenáročné. Program by ale určitě neměl důstojnou formu pro prezentaci a nedokážu si představit jak realizovat některé složitější funkce, které by měl program umět jako jsou například vyhledávání nebo přidávání záznamů.

PHP je nástavba HTML editoru, tzn., že lze říci, že můžeme uvést stejné kladné vlastnosti jako HTML, tzn. jednoduchý kód. Díky tomu, že při programování v PHP editoru již lze používat proměnné, podmínky a cykly, můžeme zde realizovat složitější funkce jako např. vyhledávání nebo přidávání záznamů. Další nespornou výhodou a zároveň nevýhodou je, že program pracuje na internetu. Tím pádem se může program jednoduše šířit a uživatel jej nemusí mít nahraný na pevném disku. Nevýhoda je, že pokud nemá uživatel právě přístup k internetu nemůže program využít a navíc také existuje mnoho různých prohlížečů a pro programátora, který nemá s PHP moc zkušeností určitě nebude jednoduché zajistit, aby program fungoval na všech prohlížečích stejně. Poslední velká přednost PHP, je, že funguje na všech operačních systémech.

Vývojové prostředí C++ Builder je poslední program, ve kterém jsem se pokusil vytvořit jakousi „demo verzi“. Program C++ Builder přirozeně umí pracovat s proměnnými, podmínkami a cykly, tudíž zde opět odpadá problém s realizováním složitějších funkcí v programu. Protože program pracuje s komponenty používanými ve Windows, má program velice příjemné a známé uživatelské prostředí, tím pádem je pro uživatele velice jednoduché si na něj zvyknout a naučit se v něm orientovat. Nevýhodou oproti editoru PHP je, že vytvořený program nebude umět pracovat na jiných operačních systémech. Díky tomu, že C++ Builder nevytváří internetovou stránku, ale „exový“ soubor, nelze výsledek práce prezentovat přímo na internetu. Určitě ale není problém na program umístit odkaz, kde se ho bude moci uživatel moci stáhnout např. ve formátu zip a potom kdykoliv pustit bez připojení na internet.

Po zvážení všech pro a proti u všech vývojových prostředích jsem se rozhodl pro vývojové prostředí C++ Builder. Poměr výhod a nevýhod mezi tímto programem a PHP je asi stejný, proto zde rozhodly moje subjektivní názory. Program C++ Builder do určité míry znám a věřím, že díky velice příjemnému uživatelskému prostředí dosáhnou lepších výsledků než při programování v PHP.

8. Vytvoření programu

8.1 Nejdůležitější funkce programu

8.1.1 Spuštění programu

Po spuštění programu, uživatel uvidí okénko, které mu dává možnost vybrat si jednu ze tří databází. Po kliknutí na obrázek nebo na nadpis databáze se zobrazí formulář, na kterém jsou funkce, které jsou popsány níže v této kapitole.



Obr 8.1.1. Hlavní okno programu

8.1.2 Funkce prohlížení

Funkce prohlížení je prioritní funkce programu, klikáním na jednotlivá témata a podtémata stromové struktury vlevo na formuláři může uživatel zobrazovat jednotlivé termíny a jejich podtermíny.

Program je vytvořen tak, že pokud uživatel klikne na skupinu termínů, tedy že nezvolil zobrazit pouze jeden termín, zobrazí se pouze texty, které jsou k daným termínům přiřazeny. Je to pouze jakýsi náhled. Obrázky a vlastní poznámky jsou pro přehlednost skryty. Zobrazí se pouze až v případě, že uživatel zvolil právě jeden termín.

8.1.3 Funkce tlačítek vpřed a zpět

Aby uživatel nemusel znovu složitě vyhledávat již zobrazené termíny, jsou v programu umístěna tlačítka vpřed a zpět. V programu je rezervováno místo pro paměť, která si pamatuje, na které termíny uživatel klikl nebo si je jiným způsobem v programu zobrazil. Velikost této paměti je pro 200 záznamů, to znamená, že uživatel může zobrazit až 200 předchozích termínů.

8.1.4 Funkce zobrazení příbuzných témat

Každému termínu se dá přiřadit až 5 příbuzných témat. Tato témata jsou zobrazena v rozbalovacím okénku nad stromovou strukturou. Po kliknutí na jedno z příbuzných témat se objeví jeho detail.

8.1.5 Funkce přidání vlastní poznámky

Ke každému termínu může uživatel přidat vlastní poznámku tak, že ji v detailu termínu napíše do textového editoru s názvem „Hier können Sie Ihr Text zugeben“. Po tom, co se stane textový editor neaktivní (kliknutí na jiný termín, ukončení programu atd.) se poznámka uloží do textového souboru se stejným názvem jako daný termín do složky, kde je program umístěn. Program se uživatele neptá, jestli chce jakkoliv změněnou poznámku uložit, je to pro to, aby program nijak uživatele zbytečně nezdržoval. Z tohoto textového souboru se také poznámka načte při dalším zobrazení daného termínu.

8.1.6 Funkce vyhledávání termínů

Funkce vyhledávání slouží k rychlému zjištění, zda databáze obsahuje hledaný termín. Pokud uživatel napíše do textového editoru nad tlačítkem „Suchen“ hledané slovo nebo jen jeho část a klikne na toto tlačítko, program zobrazí celou databázi a zkontroluje, zda se v ní toto slovo nachází. Pokud program něco najde a uživatel chce hledat dále, stačí znovu kliknout na tlačítko „Suchen“ a program hledá dále. Pokud program nic nenajde, objeví se hláška „Nicht gefunden“, což značí, že se termín nebo slovo v databázi nenachází. Funkce pro vyhledávání neukládá data do paměti pro tlačítka vpřed a zpět. Je to z toho důvodu, že funkce vyhledávání slouží jen pro rychlou orientaci v programu, ne pro práci s ním a uživatele by zdržovalo od práce, kdyby po klikání na tyto tlačítka viděl pouze výsledky toho, co už úspěšně nebo neúspěšně vyhledal.

8.1.7 Funkce tisk

V systémovém menu se nachází položka „Drucken“, která vytiskne to, co má právě uživatel zobrazeno. Pokud má uživatel zobrazen nějaký detail termínu s obrázky, vytisknou se nejprve stránky s textem, potom teprve stránky s obrázky.

8.1.8 Funkce pro zobrazení nápovědy

Položka „Hilfe“ v systémovém menu zobrazí nápovědu, která je k programu vytvořena, zde je popsáno jak česky, tak německy, jak má uživatel s programem správně pracovat.

8.1.9 Funkce přepínání mezi databázemi

Databáze můžeme přepínat dvěma způsoby, buďto otevřenou databázi ukončíme křížkem vpravo nahoře na formuláři nebo položkou v systémovém menu s názvem „Andere Dateibase“. Oba tyto způsoby uzavřou aktivní databázi a zobrazí okénko, které bylo vidět hned po spuštění programu.

8.1.10 Funkce přidávání záznamu v databázi DSP

Funkce pro přidání nových záznamů je pouze v databázi DSP. Pro další databáze je tato funkce celkem nepotřebná, protože matematika a fyzika jsou oblasti, které už se dále nerozvíjí, obsahují většinu základních pojmů a pro program nejsou prioritní. Kdyžto databáze DSP rozhodně neobsahuje všechny pojmy a digitální zpracování signálu je stále se rozvíjející obor, kde budou vznikat nové a nové pojmy.

Funkci pro přidání záznamů vyvoláme kliknutím na položku „Zugeben einen Terminus“. Program uživatele vyzve k zadání hesla, toto heslo je „0000“. Po zadání správného hesla se teprve objeví dialogové okno pro přidání termínu.

Na formuláři si uživatel může vybrat mezi dvěma jazyky a to, pochopitelně, mezi češtinou a němčinou. Volba jazyka způsobí změnu v popisech jednotlivých kolonek a tlačítek.

Formulář je rozdělen na povinnou a nepovinnou část. Povinná část je označena červenými nadpisy kolonek. Zde uživatel musí vybrat jednu z možností. Program je koncipován tak, aby musela být vždy zadána alespoň nějaká hodnota nebo aby se tyto hodnoty doplňovali správně automaticky, i když je uživatel může jakkoliv upravovat. Nepotřebné kolonky se automaticky stávají neaktivní podle zvolené kombinace předchozích hodnot. Pole pro přidávání textu je rozděleno do 3 částí. Není to sice úplně praktické a estetické, ale velmi to ulehčilo práci při programování. Do každé kolonky se vleze 255 znaků. Po napsání těchto 255 znaků kurzor automaticky přeskočí do další kolonky. Po naplnění všech tří kolonek vás program upozorní, že místo v paměti pro text je již plné a dále již nejde nic dopisovat. Kolonky nadepsané černě jsou nepovinné údaje. Jsou to kolonky pro přidání obrázků a příbuzných termínů. Program sám upozorňuje uživatele plovoucí nápovědou, jak se mají obrázky a příbuzná témata doplňovat. Obrázky se přidávají tak, že do příslušné kolonky musí uživatel napsat název obrázku ve formátu bmp bez přípony. A tento obrázek nahrát do složky „obrazky“ ve stejném adresáři jako je program. Do další kolonky musí uživatel dopsat popis, který má být pod obrázkem vidět. Pokud jednu z kolonek nevyplní, program je nastaven tak, aby obrázek nepřidal. Pokud uživatel nechce obrázek přidávat, měl by v příslušných okénkách nechat 0, pokud tak neučiní, program to udělá automaticky. Ke každému tématu je možno připojit celkem 6 obrázků. Esteticky je program navržen pro 3 schémata, tabulky nebo grafy a 3 vzorce. Jinak ale může uživatel vkládat obrázky jakkoliv bez ohledu na jejich charakter, na funkci programu to nemá vliv. Příbuzná témata se přidávají podobně jako obrázky. Do okénka uživatel musí napsat přesný název příbuzného tématu, který už v databázi je. Program totiž kontroluje, zdali se termín v databázi nachází a teprve potom dané příbuzné téma přidá. Pokud opět uživatel nechce nic přidat, měl by zde ponechat 0. Pokud tak neučiní, program to udělá za něj. Ke každému termínu lze přidat maximálně 5 příbuzných témat. Po vyplnění všech kolonek stačí už jenom kliknout na tlačítko „Přidat“ („Zugeben“) a termín se ihned objeví v databázi na posledním místě daného seznamu.

Obr 8.1.2 Okno pro přidání záznamu

8.1.11 Funkce správa přidáných záznamů

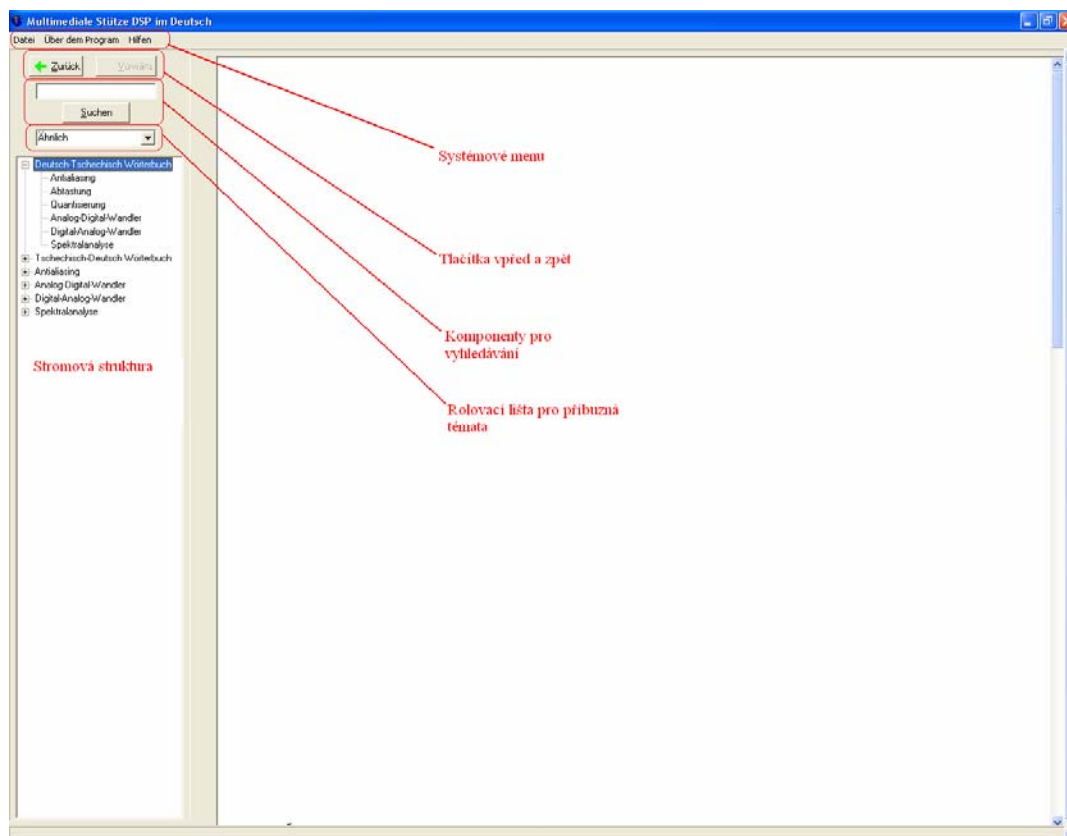
Protože přidávat záznamy lze pouze u databáze DSP, lze přidané záznamy upravovat pouze u této databáze.

Spravovat záznamy může uživatel dvěma způsoby. Buďto může přidáný záznam jakkoliv upravit nebo ho smazat. Funkci vyvoláme kliknutím na položku „Bearbeiten einen Terminus“.

Program uživatele vyzve k zadání hesla, toto heslo je „0000“. Po zadání správného hesla se teprve objeví dialogové okno pro správu termínu. Okno je téměř stejné jako pro přidání termínu, pouze s rozdílem, že vlevo dole je na okně posuvná lišta, která udává, který záznam právě upravujete.

Termíny se upravují tak, že pomocí posuvné lišty najdete termín, který chcete upravit. Poté změníte chybný nebo doplníte chybějící údaj a kliknete na tlačítko „Opravit“ („Berichtigen“). Pokud upravíte jeden nebo více termínů, program vás poté upozorní na to, že je program nutno ukončit a znova spustit. Je to z toho důvodu, že si databáze musí přeskládat svoje data. Programově to není řešeno, protože by se tím program zbytečně zpomalil a zabíral by více paměti.

Mazání termínů se dělá tak, že si pomocí posuvné lišty uživatel najde termín, který chce smazat a klikne na tlačítko „Smazat“ („Ausradieren“).



Obr. 8.1.3 Vzhled formuláře pro každou databázi

8.2 Rozbor a vysvětlení nejdůležitějších částí zdrojového kódu

Protože nejdůležitější databází programu je databáze DSP, budou funkce vysvětleny na této databázi. Ostatní databáze mají zdrojové kódy podobné, jsou v nich změněny pouze názvy proměnných a komponent na formulářích.

Celý program je, jak už bylo řečeno, v klasické verzi programu C++ Builder. I když se na prvních stránkách všude objevovalo slovo databáze, není program koncipován v žádném databázovém programovacím jazyku. Protože sem měl v zadání možnost sám si zvolit vývojové prostředí, ve kterém budu program tvořit, zvolil jsem základní verzi C++ Builder, protože jsem se chtěl dostat hlouběji do jeho problematik, naučit se, jak s programem co nejlépe pracovat, pochopit jeho problematiku a řešit v něm různé malé i velké problémy. Nakonec se ukázalo, že kvůli mým omezeným programovacím schopnostem nebyl program úplně ideální, protože se mi nepodařilo přijít na všechny závady programu. Výsledek mé práce je ale rozhodně postačující a myslím si, že i přes malé chyby se program velice dobře a jednoduše pracuje.

Myslím si, že je velice vhodné uvést, jak můj program hardwarově zaměstnává počítač. Na pevném disku program zabere necelých 27 MB, což je z velké části způsobeno velkým množstvím obrázků. Pro dnešní počítače s harddiskem větším než 80 GB je to podle mě zanedbatelná hodnota. Spuštěný program zabere maximálně 7 MB operační paměti. Pro srovnání zabere např. program ICQ 6 a Nod32 dohromady asi 60 MB operační paměti. Jak je vidět z těchto hodnot, program je vcelku jednoduchý a nenáročný.

8.2.1 Definice globálních proměnných v databázi DSP

Globální proměnné jsou proměnné definované pro celý běh programu, tzn., že při spuštění programu se jim alokuje paměť a tu zabírají po celou dobu běhu programu.

```
int x,y,z;
```

Proměnné číselného typu Integer, které slouží jako souřadnice pro zobrazování dat v databázi.

```
int res=0;
```

Proměnná číselného typu Integer, která slouží pro hlídání programu, jestli není potřeba ho restartovat.

```
int top=0,top2=0,left2,left;
```

Proměnné číselného typu Integer, které slouží pro určení polohy komponent jako textový editor, obrázek nebo popis obrázku.

```
int x1[200], y1[200], z1[200], counter1=-1, citac1=1, citac2=1, counter2=-1;
```

Proměnné číselného typu Integer, které slouží pro funkci vpřed a zpět. Pole proměnných x1, y1 a z1 udávají souřadnice předchozích zobrazených údajů. Proměnné counter1 a counter2 udávají pozici, na které se právě uživatel při procházení vpřed nebo zpět pohybuje a proměnné citac1 a citac2 udávají, o kolik pozic se uživatel posunul vzhledem k výchozí pozici. Pro funkci zpět jsou proměnné counter1 a citac1, pro funkci vpřed jsou proměnné counter2 a citac2.

```
AnsiString nazev_souboru, pripona = ".txt";
```

Proměnné znakového typu AnsiString, které slouží pro uložení a načtení poznámky poznámky.

```
int povol=0, vl_index=127, vl_index1=127, vl_index2=110;
```

Proměnné číselného typu Integer, které slouží pro přidávání vlastních záznamů. Proměnná povol udává, jestli byl do stromové struktury přidán vlastní záznam. Proměnné vl_index, vl_index1, vl_index2 udávají, na kterou pozici v paměti se mají přidávané záznamy ukládat.

```
using namespace std;
```

Proměnná, která není v programu nikde úplně využita. Je pouze potřebná pro dynamickou tvorbu komponent pro tisk.

```
AnsiString bmpName[6];
```

Pole znakového typu AnsiString, do které se nahrává název obrázku pro případný tisk.

```
AnsiString bmpCaption[6];
```

Pole znakového typu `AnsiString`, do které se nahrává popis obrázku pro případný tisk.

```
typedef struct t_termin
{
    int obor;
    int podobor;
    int poradí;
    int cislo;
    int pribuzne1;
    int pribuzne2;
    int pribuzne3;
    int pribuzne4;
    int pribuzne5;
    ShortString t_nadpis;
    ShortString t_text1;
    ShortString t_text2;
    ShortString t_text3;
} tx_termin;

t_termin *poc_terminu[280];
int count0, index;
```

Proměnná typu struktura `t_termin`, která má tvar hlavní formy použitých termínů, tedy nese jejich detail. Každý termín musí obsahovat číslo oboru, podoboru a pořadí. Dále pak číslo, které je použito při hledání příbuzných témat společně s pěti příbuznými čísly pro přiřazení příbuzného tématu. Název termínu je uložen v proměnné `t_nadpis`. Pro popis termínu je alokována paměť v proměnných `t_text1` až `t_text3`. Tento ne úplně ideální způsob je použit z toho důvodu, že do struktury nelze přiřadit proměnná typu `AnsiString`.

Společně s touto strukturou je nutno uvést ještě také ukazatel `poc_terminu`, kterému je alokována stejná velikost jako struktuře `t_termin`. Číslo 280 udává, že databáze je udělaná pro 280 záznamů, po té už do databáze žádné termíny přidávat nejdou. Tento řetězec slouží pro uložení jednotlivých termínů do paměti programu. Proměnná typu `Integer` `count0` udává počet uložených termínů a proměnná typu `Integer` `index` udává, přesnou pozici uloženého záznamu.

```
typedef struct t_seznamy //struktura pro zobrazení slovníku
{
    int obor;
    int podobor;
    ShortString t_nadpis1;
    ShortString t_nadpis2;
    ShortString t_text1;
    ShortString t_text2;
} tx_seznamy;

t_seznamy *poc_seznamu[280];
int count1=0, index1;
```

Proměnná typu struktura `t_seznamy`, která slouží k zobrazování slovníků. Každý termín musí obsahovat číslo oboru, podoboru a pořadí. Dále pak německý a český název skupiny, pod kterou daný záznam patří `t_nadpis1` a `t_nadpis2` a německý a český název daného termínu `t_text1` a `t_text2`.

Společně s touto strukturou je nutno uvést ještě také ukazatel `poc_seznamu`, kterému je alokována stejná velikost jako struktuře `t_seznamy`. Číslo 280 opět udává, že databáze je udělaná pro 280 záznamů, po té už do databáze žádné termíny přidávat nejdou. Tento řetězec slouží pro uložení jednotlivých termínů do paměti programu. Proměnná typu `Integer` `count1` udává počet uložených termínů a proměnná typu `Integer` `index1` udává, přesnou pozici uloženého záznamu.

```
typedef struct t_obraz
{
    int obor;
    int podobor;
    int poradi;
    int cislo;
    int druh;
    ShortString t_nazev;
    ShortString t_cesta;
} tx_obraz;

t_obraz *poc_obrazu[1032];
int count2=0,index2=0;
```

Proměnná typu struktura `t_obraz`, která slouží k zobrazování obrázků. Každý termín musí obsahovat číslo oboru, podoboru a pořadí. Dále pak číslo pro zobrazení obrázků ve správném pořadí, obsahuje-li termín více obrázků najednou a druh, který udává, jedná-li se o obrázek či vzoreček. Proměnné `t_nazev` a `t_cesta` udávají popis k obrázku a cestu, odkud se má obrázek načíst.

Společně s touto strukturou je nutno uvést ještě také ukazatel `poc_obrazu`, kterému je alokována stejná velikost jako struktuře `t_obraz`. Číslo 1032 udává, že databáze je udělaná pro 280 záznamů (každému termínu lze přiřadit 6 obrázků), po té už do databáze žádné termíny přidávat nejdou. Tento řetězec slouží pro uložení jednotlivých termínů do paměti programu. Proměnná typu Integer `count2` udává počet uložených termínů a proměnná typu Integer `index2` udává, přesnou pozici uloženého záznamu.

8.3 Vysvětlení funkcí, které jsou v programu volány

8.3.1 Funkce `adding()` pro přidání nového záznamu `t_termín`

```
t_termín *adding(int obor, int podobor,int poradi, int cislo,int
pribuzne1,int pribuzne2,int pribuzne3,int pribuzne4,int
pribuzne5,ShortString t_nadpis, ShortString t_text1, ShortString t_text2,
ShortString t_text3)
{t_termín* termin;
termin=(t_termín*)malloc(sizeof(t_termín));
termin->obor=obor;
termin->podobor=podobor;
termin->poradi=poradi;
termin->cislo=cislo;
termin->pribuzne1=pribuzne1;
termin->pribuzne2=pribuzne2;
termin->pribuzne3=pribuzne3;
termin->pribuzne4=pribuzne4;
termin->pribuzne5=pribuzne5;
termin->t_nadpis=t_nadpis;
termin->t_text1=t_text1;
termin->t_text2=t_text2;
termin->t_text3=t_text3;
return termin;}
```

Funkce tohoto typu jsou v programu tři, jedna pro detail termínu, druhá pro slovník a třetí pro obrázky. Funkce má stejné vstupní hodnoty jako struktura `t_termín`. Je potřeba opět uvolnit místo v paměti, k tomu slouží ukazatel `termin` a funkce `malloc()`. Potom už se jen každému parametru struktury `t_termín` přiřadí správné hodnoty.

Příklad pro přidání nového záznamu může vypadat takto:

```
poc_terminu[0]=adding(1,1,1,1,0,0,0,0,0, "Nadpis terminu","První část textu","Druhá část textu","Třetí část textu")
```

8.3.2 Funkce show(), která říká, jak zobrazovat detaily jednotlivých termínů

```
void show (int index)
{
    top=10, left=230;
    f_hlavni->re_text->Top=top;
    f_hlavni->re_text->Left=left;
    AnsiString nadpis;
    AnsiString mezera;
    AnsiString text;
    nadpis+=" "+poc_terminu[index]->t_nadpis+" ";
    f_hlavni->re_text->SelAttributes->Size=13;
    f_hlavni->re_text->Lines->Add(nadpis);
    text+=poc_terminu[index]->t_text1;
    text+=poc_terminu[index]->t_text2;
    text+=poc_terminu[index]->t_text3;
    f_hlavni->re_text->Lines->Add(mezera);
    f_hlavni->re_text->SelAttributes->Size=10;
    f_hlavni->re_text->Lines->Add(text);
    f_hlavni->re_text->Lines->Add(mezera);
}
```

Tato funkce má vstupní parametr index, který udává, se kterým záznamem se má pracovat.

Nejprve se nastaví poloha textového editoru re_text pomocí proměnných top a left

Dále se nadefinují lokální proměnné nadpis, mezera a text. Do proměnné nadpis se nakopíruje proměnná t_nadpis příslušného záznamu a zobrazí se do textového editoru s předem nadefinovaným fontem. Do proměnné text se podobně nakopírují proměnné t_text1 až t_text3 a zobrazí se do textového editoru s předem nadefinovaným fontem. Mezi jednotlivými záznamy se vytvoří mezery pomocí proměnné mezera.

8.3.3 Funkce show1(), která říká, jak zobrazovat německo-český slovník

```
void show1 (int index1)
{
    AnsiString text;
    text+=" "+poc_seznamu[index1]->t_text1;
    text+="\t";
    text+=" "+poc_seznamu[index1]->t_text2;
    f_hlavni->re_text->SelAttributes->Size=10;
    f_hlavni->re_text->Lines->Add(text);
}
```

Tato funkce má vstupní parametr index1, který udává, se kterým záznamem se má pracovat.

Dále se nadefinuje lokální proměnná text. Do této proměnné se nahraje nejprve proměnná t_text1 zpracovávaného termínu, pak se zde vloží tabulátor a proměnná t_text2 zpracovávaného termínu. Textovému editoru se nastaví font a zobrazí se do něj proměnná text.

8.3.4 Funkce show2(), která říká, jak zobrazovat česko-německý slovník

```

void show2 (int index1)
{
    AnsiString text;
    text+=" " + poc_seznamu[index1]->t_text2;
    text+="\t\t";
    text+=" " + poc_seznamu[index1]->t_text1;
    f_hlavni->re_text->SelAttributes->Size=10;
    f_hlavni->re_text->Lines->Add(text);
}

```

Tato funkce má vstupní parametr index1, který udává, se kterým záznamem se má pracovat.

Dále se nadefinuje lokální proměnná text. Do této proměnné se nahraje nejprve proměnná t_text2 zpracovávaného termínu, pak se zde vloží tabulátor a proměnná t_text1 zpracovávaného termínu. Textovému editoru se nastaví font a zobrazí se do něj proměnná text.

8.3.5 Funkce showimg(), která říká, jak zobrazovat obrázky

```

void showimg(int index2)
{
    AnsiString popis;
    AnsiString popis2;
    popis+=poc_obrazu[index2]->t_nazev;
    popis2+=poc_obrazu[index2]->t_nazev;
    if (poc_obrazu[index2]->druh<4&& poc_obrazu[index2]->cislo!=0)
    {
        if (poc_obrazu[index2]->druh==1)
        {
            left=230;
            top+=20+f_hlavni->re_vlastni->Height+f_hlavni->re_text->Height;
            f_hlavni->i_obraz1->Top=top;
            f_hlavni->i_obraz1->Left=left;
            try {f_hlavni->i_obraz1->Picture->LoadFromFile(poc_obrazu[index2]
                ->t_cesta);
                bmpName[0]=poc_obrazu[index2]->t_cesta;
                bmpCaption[0]=poc_obrazu[index2]->t_nazev;
                f_hlavni->i_obraz1->Width=f_hlavni->i_obraz1->Picture->Bitmap->Width;
                hlavni->i_obraz1->Height=f_hlavni->i_obraz1->Picture->Bitmap->Height;}
            catch (EFOpenError &e) {MessageDlg("Es fällt Ordner "
                +poc_obrazu[index2]->t_cesta+"!", mtError, TMsgDlgButtons()<<mbOK,0);}
            top+=10+f_hlavni->i_obraz1->Height;
            f_hlavni->l_popis1->Top=top;
            f_hlavni->l_popis1->Left=left;
            f_hlavni->l_popis1->Caption=popis;
            top+=10+f_hlavni->l_popis1->Height;}}
    if (poc_obrazu[index2]->druh>=4&& poc_obrazu[index2]->cislo!=0)
    {
        if (poc_obrazu[index2]->druh==4)
        {
            if (top==0)
            {
                top2+=20+f_hlavni->re_vlastni->Height+f_hlavni->re_text->Height;}
            else
            {
                top2=top;}
            f_hlavni->i_obraz4->Top=top2;
            f_hlavni->i_obraz4->Left=left;
            try {f_hlavni->i_obraz4->Picture->LoadFromFile(poc_obrazu[index2]
                ->t_cesta);
                bmpName[3]=poc_obrazu[index2]->t_cesta;
                bmpCaption[3]=poc_obrazu[index2]->t_nazev;
                f_hlavni->i_obraz4->Width=f_hlavni->i_obraz4->Picture->Bitmap->Width;
                f_hlavni->i_obraz4->Height=f_hlavni->i_obraz4->Picture->Bitmap
                ->Height;}
            catch (EFOpenError &e) {MessageDlg("Es fällt Ordner "
                +poc_obrazu[index2]->t_cesta+"!", mtError, TMsgDlgButtons()<<mbOK,0);}
            top2+=10+f_hlavni->i_obraz4->Height;
            f_hlavni->l_popis4->Top=top2;
        }
    }
}

```

```
f_hlavni->l_popis4->Left=left;
f_hlavni->l_popis4->Caption=popis2;
top2+=10+f_hlavni->l_popis4->Height;}}
```

V ukázce je funkce zkrácena na to, že ke každému termínu lze přiřadit pouze jeden obrázek a jeden vzorec. Vstupním parametrem této funkce je proměnná index2, která udává, se kterým záznamem se má pracovat.

Nejprve se nadefinují lokální proměnné popis a popis2, které slouží pro popisy, které mají být pod obrázkem vidět. Do těchto proměnných se načte popis ze struktury t_obraz. O tom, která z proměnných se použije, se dále rozhodne podle podmínek.

Při zobrazování se nejprve program podívá, jestli je proměnná číslo daného obrázku různá od 0, tzn., jestli existuje a jestli je menší než 4, tedy jestli se jedná o obrázek. Pokud ano, zkontroluje program, jestli se jedná o první, druhý nebo třetí obrázek. (V příkladu je pouze podmínka pro první obrázek.) Po rozhodnutí, kolikátý obrázek je se nastaví proměnná left na konstantní hodnotu 230 a proměnná top na hodnotu šířky textového pole pro detail termínu + šířky textového pole pro poznámku + šířky všech předchozích obrázků a jejich popisků + konstanta 20. Tyto dvě hodnoty se nastaví jako výchozí poloha pro zobrazení obrázku. Program se pokusí načíst obrázek, který má název jako hodnota uložená v proměnné t_cesta, zpracovávaného termínu struktury t_obraz. Nastaví hodnotu proměnných bmpName a bmpCaption pro případný tisk a zvětší plochu pro zobrazení obrázku na jeho velikost a zobrazí ho. Pokud se programu nepodaří obrázek načíst, objeví se chybová hláška, že obrázek nebyl nalezen. Po zobrazení obrázku se podobným způsobem načte i jeho popis.

Pokud je proměnná číslo daného termínu větší a rovno 4, jedná se o vzorec. Program nejprve zkontroluje, jestli je k termínu přiřazen nějaký obrázek. To zjistí podle proměnné top, která bude různá od nuly. Pokud je proměnná top 0, nastaví počáteční souřadnice plochy pro zobrazení těsně pod textové editory. Pokud je top různá od nuly nastaví program souřadnice plátna těsně pod poslední popis obrázku. Potom už program pracuje stejně, jako když se jednalo o obrázek a ne o vzorec.

8.3.6 Funkce termíny(), která označuje termíny databáze v textu

```
void termíny (int index)
{int pozice1, zacatek1, do_koncel1,a=0;
f_hlavni->e_hledej->Text=poc_termínu[index]->t_nadpis;a=0;
do
{if (f_hlavni->re_text->SelLength!=0)
    zacatek1 = f_hlavni->re_text->SelStart + f_hlavni->re_text->SelLength;
else
    {zacatek1 = 0;}

do_koncel1 = f_hlavni->re_text->Text.Length() - zacatek1;
pozice1 = f_hlavni->re_text->FindText(f_hlavni->e_hledej->Text,
zacatek1, do_koncel1, TSearchTypes() << stMatchCase);
if (pozice1 != -1)
    {f_hlavni->re_text->SetFocus();
    f_hlavni->re_text->SelStart = pozice1;
    f_hlavni->re_text->SelLength = f_hlavni->e_hledej->Text.Length();
    f_hlavni->re_text->SelAttributes->Color=clGreen}
else
    {f_hlavni->re_text->SelLength=!f_hlavni->re_text->SelLength;
    a=1;}}
while (a==0);
f_hlavni->e_hledej->Clear();
f_hlavni->e_hledej->SetFocus();}
```

Vstupním parametrem této funkce je proměnná `index`. Funkce pro označení termínů pracuje tak, že program v zobrazeném textu vyhledá všechny termíny, které se v databázi nacházejí a označí je zeleně. Pro tuto funkci je použita pomocná komponenta `e_hledej`. Do které se načítají vyhledávané termíny.

Nejprve se nadefinují lokální proměnné pozice pro určení pozice kurzoru, začátek pro určení počátku vyhledávání, `do_konce`, která určuje délku do konce textu a proměnná `a`, která oznamuje funkci konec vyhledávání.

Program dělá jeden úkon tak dlouho, dokud vyhovuje podmínka, že bylo něco před tím nalezeno. Při vyhledávání program nejdříve zkontroluje, jestli už bylo něco před tím nalezeno. Pokud ano, nastaví zaměření pro začátek vyhledávání a místo, kde naposledy něco našel. Pokud nic před tím nalezeno nebylo, program nastaví začátek na pozici nula.

Následně program určí délku textu do konce a zahájí samotné vyhledávání funkcí `FindText` s příslušnými parametry.

Další podmínka programu říká, že pokud bylo něco nalezeno, nastav zaměření do textového editoru, řekni programu, od jakého místa má vyhledávat dále, označ nalezený text a změň ho na zelený. A pokud se nic nenašlo, řekni to programu.

Po skončení vyhledávání se vyčistí textový editor `e_hledej`, který při vyhledávání pomáhal místo další definované proměnné.

8.3.7 Funkce `AddNode()`, která přidává záznam do stromové struktury

```
bool AddNode(TTreeNode *n, AnsiString src, AnsiString trg)
{ if (n->Text==src&&pvool==0
    {pvool=1;
      f_hlavni->t_seznam->Items->AddChild(n,trg)
    return true;}}
```

Tato funkce má vstupní parametry třídu `TTreeNode` `n`, která říká jak se stromovou strukturou pracovat, proměnnou `src`, která udává, kam se má záznam do stromové struktury přidat a proměnnou `trg`, která udává název toho, co se má do struktury přidat.

Funkce `AddNode` říká, že pokud ve stromové struktuře existuje nějaký obor, který je stejný jako obor přidávaného a je zakázáno přidávat záznam, povol to a přidej záznam se jménem v proměnné `trg`.

Na konci se vrátí s logická hodnota `true`, která je potřebná pro správné vyhodnocení funkce v programu, která zavolala tuto podfunkci.

U této funkce se při překládání objevuje varování, že funkce nevrací žádnou hodnotu. Tento problém jde sice vyřešit, ale program potom nepracuje úplně správně. Toto varování nemá na funkce programu vliv.

8.3.8 Funkce `uloz()` pro uložení přidáných záznamů do textového souboru

K tomu, aby se mohly přidaná data načíst při dalším spuštění programu je třeba je uložit do textového souboru

```
void uloz (void)
{DeleteFile("databaze1.txt");
 TStringList* to_file = new TStringList();
 to_file->Add(IntToStr(vl_index));
 for(int i=127; i<vl_index; i++)
 {to_file->Add(IntToStr(poc_terminu[i]->obor));
  to_file->Add(IntToStr(poc_terminu[i]->podobor));
  to_file->Add(IntToStr(poc_terminu[i]->poradi));
  to_file->Add(IntToStr(poc_terminu[i]->cislo));
 }
```

```

to_file->Add(IntToStr(poc_terminu[i]->pribuzne1));
to_file->Add(IntToStr(poc_terminu[i]->pribuzne2));
to_file->Add(IntToStr(poc_terminu[i]->pribuzne3));
to_file->Add(IntToStr(poc_terminu[i]->pribuzne4));
to_file->Add(IntToStr(poc_terminu[i]->pribuzne5));
to_file->Add(poc_terminu[i]->t_nadpis);
to_file->Add(poc_terminu[i]->t_text1);
to_file->Add(poc_terminu[i]->t_text2);
to_file->Add(poc_terminu[i]->t_text3);}
to_file->SaveToFile("databazel.txt");

```

Funkce nemá žádný vstupní parametr, pro jednoduchost je zde uvedena pouze funkce pro ukládání obdoby struktury `t_termin`. Ostatní struktury se ukládají analogicky.

Nejprve je vymazán předchozí soubor. Dále program vytvoří novou třídu `to_file`, která umí ukládat text do textového souboru. Na první řádek uloží program aktuální velikost proměnné `vl_index`, aby program při načítání věděl, kolik přidanych záznamů má opět načíst. Poté program uloží všechny přidané termíny, tak že každá proměnná daného záznamu je na jednom řádku, což je výhodné pro načítání. Nakonec se program uloží třídu `to:file` do textového editoru s názvem `databazel.txt`.

8.3.8 Funkce `reset()`, která „čistí“ komponenty pro zobrazení dalšího záznamu

```

void reset()
{
    f_hlavni->re_text->Clear();
    f_hlavni->l_popis1->Caption="";
    f_hlavni->l_popis2->Caption="";
    f_hlavni->l_popis3->Caption="";
    f_hlavni->l_popis4->Caption="";
    f_hlavni->l_popis5->Caption="";
    f_hlavni->l_popis6->Caption="";
    f_hlavni->i_obraz1->Picture=0;
    f_hlavni->i_obraz2->Picture=0;
    f_hlavni->i_obraz3->Picture=0;
    f_hlavni->i_obraz4->Picture=0;
    f_hlavni->i_obraz5->Picture=0;
    f_hlavni->i_obraz6->Picture=0;}

```

Tato jednoduchá funkce před tím, než se zobrazí další termíny, vyčistí všechny potřebné komponenty.

8.3.9 Funkce `zobraz()` pro zobrazení vybraných termínů programem

```

void zobraz(void)
{
    if (x==1)
    {
        f_hlavni->re_text->Clear();
        f_hlavni->re_text->Width=f_hlavni->Width-270;
        f_hlavni->re_text->Height=f_hlavni->Height-75;
        f_hlavni->re_vlastni->Visible=false;
        int j=0,i=0;
        for (index1=0; index1<count1; index1++)// pro vsechny termíny
        {
            if ((poc_seznamu[index1]->obor==1)&&(y==0))
            {
                if (poc_seznamu[index1]->podobor>i)
                {
                    f_hlavni->re_text->SelAttributes->Size=13;
                    f_hlavni->re_text->Lines->Add("\n "+poc_seznamu[index1]
                    ->t_nadpis1+"\n");
                    i=poc_seznamu[index1]->podobor;}
                if(index1>=127&&j==0)
                {j=1;

```

```

        f_hlavni->re_text->SelAttributes->Size=13;
        f_hlavni->re_text->Lines->Add("\n Eigenstrom\n");}
        show1(index1);}
if ((poc_seznamu[index1]->obor==1)&&(y!=0)&&(poc_seznamu[index1]
->podobor==y))
{if (poc_seznamu[index1]->podobor>i)
{f_hlavni->re_text->SelAttributes->Size=13;
f_hlavni->re_text->Lines->Add("\n "+poc_seznamu[index1]
->t_nadpis1+"\n");
i=poc_seznamu[index1]->podobor;}
if(index1>=127&&j==0)
{j=1;//nadpis se zobrazí jednou
f_hlavni->re_text->SelAttributes->Size=13;
f_hlavni->re_text->Lines->Add("\n Eigenstrom\n");}
show1(index1);}}}
if (x==3)
f_hlavni->re_text->Clear();
f_hlavni->re_text->Width=f_hlavni->Width-270;
f_hlavni->re_text->Height=f_hlavni->Height-75;
for (index=0; index<count0; index++)
{if ((poc_terminu[index]->obor==x)&&(y==0))
{show(index);
f_hlavni->re_vlastni->Visible=false;}
if ((poc_terminu[index]->obor==x)&&(y!=0)&&(poc_terminu[index]
->podobor==y))
{f_hlavni->re_text->Width=f_hlavni->Width-270;
f_hlavni->re_text->Height=f_hlavni->Height-550;
f_hlavni->re_vlastni->Top=f_hlavni->re_text->Height;
f_hlavni->re_vlastni->Left=left;
f_hlavni->re_vlastni->Width=f_hlavni->re_text->Width;
show(index);
f_hlavni->re_vlastni->Visible=true;
nazev_souboru=poc_terminu[index]->t_nadpis;
try {f_hlavni->re_vlastni->Lines->LoadFromFile(nazev_souboru +
pripona);}
catch (EOpenError &E) {f_hlavni->re_vlastni->Lines-> LoadFromFile
("chyba.txt");}
f_hlavni->c_pribuzne->Clear();
int pribuzny;
pribuzny=poc_terminu[index]->pribuzne1;
if (pribuzny!=0)
f_hlavni->c_pribuzne->AddItem(poc_terminu[pribuzny-1]
->t_nadpis,f_hlavni);
pribuzny=poc_terminu[index]->pribuzne2;
if (pribuzny!=0)
f_hlavni->c_pribuzne->AddItem(poc_terminu[pribuzny-1]
->t_nadpis,f_hlavni);
pribuzny=poc_terminu[index]->pribuzne3;
if (pribuzny!=0)
f_hlavni->c_pribuzne->AddItem(poc_terminu[pribuzny-1]
->t_nadpis,f_hlavni);
pribuzny=poc_terminu[index]->pribuzne4;
if (pribuzny!=0)
f_hlavni->c_pribuzne->AddItem(poc_terminu[pribuzny-1]
->t_nadpis,f_hlavni);
pribuzny=poc_terminu[index]->pribuzne5;
if (pribuzny!=0)
f_hlavni->c_pribuzne->AddItem(poc_terminu[pribuzny-1]
->t_nadpis,f_hlavni);}}// pridej tyto pribuzne do comba
top=0; top2=0;
bmpName[0]="0"; bmpCaption[0]="0"; bmpName[1]="0";

```

```

        bmpCaption[1]="0"; bmpName[2]="0"; bmpCaption[2]="0";
        bmpName[3]="0"; bmpCaption[3]="0"; bmpName[4]="0";
        bmpCaption[4]="0"; bmpName[5]="0"; bmpCaption[5]="0";
    for (index2=0; index2<count2; index2++)
        {if (poc_obrazu[index2]->obor==x&&y!=0&&poc_obrazu[index2]
        ->podobor==y)
            {showimg(index2);}}}
try {for (index=0; index<count0; index++)
    {if (x!=1&&x!=2)
        terminy(index);}
    for (int i=0; i<5; i++)
        {abeceda();}}
catch (EInvalidOperation &A){;}

```

Tato funkce je asi nejdůležitější funkce programu. V příkladu není opět uvedena celá. Dá se rozdělit do tří částí. První část je část pro zobrazení slovníků, druhá část je pro zobrazení detailu termínu a třetí část je volání funkce pro označení důležitých termínů zelenou barvou.

Podle hodnoty souřadnice x, která se mění při každé změně stromové struktury, se program dostane do jedné z podmínek, pokud je x rovno hodnotě 1 nebo 2, jedná se o slovníky, pokud je x větší jedná se o zobrazení detailů termínu. Pokud tedy vybereme slovník, vyčistí se nejprve textový editor a nastaví se jeho velikost, aby byl téměř přes celou obrazovku. Textový editor pro poznámku zmizí. Dále je zde podmínka, jestli se má zobrazit úplně celý seznam nebo pouze jeho část. Program to dělá pomocí souřadnice y. Jestli je y rovno nule, zobrazujeme celý seznam. Pokud ne, zobrazujeme pouze jeho část. Dále je potřeba zobrazit jednotlivé nadpisy oborů. To je uskutečněno pomocí konstanty i, konstanta i se nastaví na nulu a testuje se, jestli je obor zobrazovaného termínu větší než tato konstanta. Pokud ano, zobrazí se nadpis a k i se přičte jedna. Tím se zabezpečí zobrazení nadpisu pouze jednou. Pokud se v databázi objevují přidané záznamy, je potřeba je také oddělit nadpisem. Vygenerování nadpisu je zabezpečeno pomocí konstanty j, která se nejprve nastaví na nulu. Poté je programem testováno, jestli se v databázi vyskytují přidané záznamy a je-li j stále rovno nule. Pokud ano, nadpis se zobrazí.

Zobrazení detailu, tedy je-li x větší než 2 se provádí podobným způsobem jako zobrazení slovníku. Nejprve se vyčistí textový editor a jeho velikost se nastaví přes celou obrazovku. Dále podmínka testuje, jedná-li se o detail jednoho nebo více termínů. Pokud se jedná o detail více termínů, je funkce jednoduchá a spočívá pouze v zavolání funkce show() pro termíny, které vyhovují podmínce a skrytí textového editoru pro poznámku. Pokud se jedná o zobrazení jednoho termínu, je funkce složitější. Nejprve se vymaže textový editor a zmenší se jeho velikost. Pak se zobrazí editor pro poznámku. Dále program testuje, nebyla-li k danému termínu už někdy dříve přidaná nějaká poznámka. Pokud ano, program ji načte z textového editoru a vepíše to editoru pro poznámku. Další krok spočívá v tom zobrazit do BomboBoxu pro výběr příbuzného termínu příbuzná témata. Program tento BomboBox nejprve vyčistí. Dále pak postupně načítá do lokální proměnné příbuzní proměnné z struktury t_terminu pribuzne1 až pribuzne5. Pokud se hodnota v těchto proměnných nerovná nule, tzn., že příbuzné téma existuje, zobrazí do KomboBoxu název termínu pod indexem, který tato proměnná obsahuje zmenšená o jedničku. Dále pak už funkce obsahuje pouze vymazání některých konstant pro další práci s programem a zobrazení obrázků daného termínu. Podmínka pro zobrazení říká, pokud má nějaký obrázek stejné souřadnice jako zobrazený termín, zobraz i tyto obrázky.

Poslední část je volání funkce označení vybraných termínů zelenou barvou a zobrazení řecké abecedy. Toto volání musí být ošetřeno jako Run-Time chyba, protože funkce zobraz je volána při každé změně okna, což znamená i při ukončování aplikace a program, který je ukončován nemůže volat další funkci. Pokud se tedy nejedná o ukončování aplikace. Program

ověří podmínku, jestli nebyl zobrazen seznam (zde by bylo vybarveno zeleně vše). A pro všechny termíny v databázi zavolá funkce termíny.

8.3.10 Funkce nacti(), která načte do databáze přidané termíny

```
void nacti (void)
{try {TStringList *from_file = new TStringList();
    t_termin *a_termin;
    from_file->LoadFromFile("databaze1.txt");
    count0 = ( vl_index = StrToInt( from_file->Strings[0]));
    for(int i=127; i<vl_index; i++)
    {free(poc_terminu[i]);
     a_termin = (t_termin *) malloc(sizeof(t_termin));
     a_termin->obor = StrToInt(from_file->Strings[13*(i-127)+1]);
     a_termin->podobor = StrToInt(from_file->Strings[13*(i-127)+2]);
     a_termin->poradi = StrToInt(from_file->Strings[13*(i-127)+3]);
     a_termin->cislo = StrToInt(from_file->Strings[13*(i-127)+4]);
     a_termin->pribuzne1 = StrToInt(from_file->Strings[13*(i-127)+5]);
     a_termin->pribuzne2 = StrToInt(from_file->Strings[13*(i-127)+6]);
     a_termin->pribuzne3 = StrToInt(from_file->Strings[13*(i-127)+7]);
     a_termin->pribuzne4 = StrToInt(from_file->Strings[13*(i-127)+8]);
     a_termin->pribuzne5 = StrToInt(from_file->Strings[13*(i-127)+9]);
     a_termin->t_nadpis = from_file->Strings[13*(i-127)+10];
     a_termin->t_text1 = from_file->Strings[13*(i-127)+11];
     a_termin->t_text2 = from_file->Strings[13*(i-127)+12];
     a_termin->t_text3 = from_file->Strings[13*(i-127)+13];
     poc_terminu[i] = a_termin;
     povol=0;
     for(int j=14;j<f_hlavni->t_seznam->Items->Count;j
{if (poc_terminu[i]->obor!=4)
    if(AddNode2(f_hlavni->t_seznam->Items->Item[j],poc_terminu[i]
->obor,poc_terminu[i]->t_nadpis))
        if (f_hlavni->t_seznam->Items->Item[j]->ImageIndex==0)
            f_hlavni->t_seznam->Items->Item[j]->ImageIndex=poc_terminu[i]->obor;
            f_hlavni->t_seznam->Items->Item[j]->SelectedIndex=poc_terminu[i]
->podobor;
            f_hlavni->t_seznam->Items->Item[j]->StateIndex=poc_terminu[i]
->poradi;}}}}
if (poc_terminu[i]->obor==4)
    {if(AddNode3(f_hlavni->t_seznam->Items->Item[j],poc_terminu[i]
->podobor,poc_terminu[i]->t_nadpis))
    { if (f_hlavni->t_seznam->Items->Item[j]->ImageIndex==0)
        {f_hlavni->t_seznam->Items->Item[j]->ImageIndex=poc_terminu[i]
->obor;
         f_hlavni->t_seznam->Items->Item[j]->SelectedIndex=poc_terminu[i]
->podobor;
         f_hlavni->t_seznam->Items->Item[j]->StateIndex=poc_terminu[i]
->poradi;}}}}}}
catch (EFOpenError &E){;}
povol=0;}
```

Funkce je potřebná pro načtení termínů do databáze z textového souboru při spuštění programu. Funkce zde není uvedena celá, je zde uvedena pouze část, která načítá data pro strukturu t_termin. Načtení dalších struktur je obdobné. Celá funkce je ošetřena proti Run-Time chybě, která ošetřuje to, že neexistuje textový soubor, ze kterého se mají data načítat.

Nejprve program vytvoří novou třídu from_file, která slouží pro uložení termínů do paměti. Do této paměti se načtou data z textového souboru databaze1.txt. Na prvním řádku je uložen počet záznamů. Tento záznam se převede z textové podoby na číslo a nahraje do

proměnných `vl_index` a `count0`. Dále se načtou všechny data ze souboru pomocí nové struktury `a_termín`. Protože mají všechny data stejnou strukturu. Může se s výhodou využít periodického opakování řádků, tzn., že se dal určit matematický předpis, podle kterého můžeme data z textového souboru načítat.

Dále je potřeba zobrazit záznamy ve stromové struktuře. K zobrazení jsou určeny funkce `AddNode2` a `AddNode3`. O jejich použití rozhodne charakter přidávaného termínu.

Syntaxe funkcí `AddNode2` a `AddNode3` je stejná jako syntaxe funkce `AddNode`. Liší se pouze tím, která data funkce zpracovává. Nejprve se tedy zakáže přidávat nody. Potom se pro všechny termíny provede podmínka, která určí jejich formu. Podle podmínky se zavolá jedna z funkcí `AddNode2` nebo `AddNode3`. Po přidání záznamu do struktury se mu ještě přiřadí souřadnice stejné jako má termín, který má záznam ukazovat.

8.3.11 Funkce `zobraz2()`, která zobrazuje termín na formuláři pro správu termínů

```
void zobraz2 (void)
{int j=0;
int cislo_indexu=f_sprava->sb_cislo_indexu->Position;
AnsiString prib1="0", prib2="0", prib3="0", prib4="0", prib5="0";
for (int index=0; index<count0; index++)
{int pr_cislo=poc_terminu[index]->cislo;
if (pr_cislo==poc_terminu[cislo_indexu]->pribuzne1)
prib1=poc_terminu[index]->t_nadpis;
if (pr_cislo==poc_terminu[cislo_indexu]->pribuzne2)
prib2=poc_terminu[index]->t_nadpis;
if (pr_cislo==poc_terminu[cislo_indexu]->pribuzne3)
prib3=poc_terminu[index]->t_nadpis;
if (pr_cislo==poc_terminu[cislo_indexu]->pribuzne4)
prib4=poc_terminu[index]->t_nadpis;
if (pr_cislo==poc_terminu[cislo_indexu]->pribuzne5)
prib5=poc_terminu[index]->t_nadpis;}

if (poc_terminu[cislo_indexu]->obor==3||poc_terminu[cislo_indexu]
->obor==5||poc_terminu[cislo_indexu]->obor==6)
{f_sprava->c_obor->ItemIndex=poc_terminu[cislo_indexu]->obor-3;
f_sprava->e_podobor->Text=IntToStr(poc_terminu[cislo_indexu]->podobor);
f_sprava->c_podobor->Enabled=false;
f_sprava->e_poradi->Text="1";
f_sprava->e_poradi->Enabled=false;}
if (poc_terminu[cislo_indexu]->obor==4)
{f_sprava->c_obor->ItemIndex=poc_terminu[cislo_indexu]->obor-3;
f_sprava->c_podobor->ItemIndex=poc_terminu[cislo_indexu]->podobor-1;
f_sprava->e_podobor->Enabled=false;
f_sprava->e_poradi->Text=IntToStr(poc_terminu[cislo_indexu]->poradi);}
f_sprava->c_clen->Text=poc_seznamu[cislo_indexu]->t_text1;
for(int i=0;i<f_sprava->c_clen->Text.Length();i++)
{f_sprava->c_clen->SelStart=i;
f_sprava->c_clen->SelLength=1;
if (f_sprava->c_clen->SelText!="")
{j++;}
else
{i=f_sprava->c_clen->Text.Length();}}
f_sprava->c_clen->SelStart=j+2;
f_sprava->c_clen->SelLength=1;
f_sprava->c_clen->Text=f_sprava->c_clen->SelText;
j=0;
f_sprava->e_nem_nazev->Text=poc_terminu[cislo_indexu]->t_nadpis;
f_sprava->e_ces_nazev->Text=poc_seznamu[cislo_indexu]->t_text2;
```

```

for(int i=3;i<f_sprava->e_ces_nazev->Text.Length();i++)
{f_sprava->e_ces_nazev->SelStart=i;
f_sprava->e_ces_nazev->SelLength=i+1;
if (f_sprava->e_ces_nazev->SelText!="\t")
j++;}
f_sprava->e_ces_nazev->SelStart=0;
f_sprava->e_ces_nazev->SelLength=j;
f_sprava->e_ces_nazev->Text=f_sprava->e_ces_nazev->SelText;
f_sprava->e_pr_text1->Text=poc_termínu[cislo_indexu]->t_text1;
f_sprava->e_pr_text2->Text=poc_termínu[cislo_indexu]->t_text2;
f_sprava->e_pr_text3->Text=poc_termínu[cislo_indexu]->t_text3;
f_sprava->e_pribuzne1->Text=prib1;
f_sprava->e_pribuzne2->Text=prib2;
f_sprava->e_pribuzne3->Text=prib3;
f_sprava->e_pribuzne4->Text=prib4;
f_sprava->e_pribuzne5->Text=prib5;

for (int vl_index2=110;vl_index2<count2;vl_index2++)
{if (poc_termínu[cislo_indexu]->obor==poc_obrazu[vl_index2]
->obor&& poc_termínu[cislo_indexu]->podobor==poc_obrazu[vl_index2]
->podobor&& poc_termínu[cislo_indexu]->poradi==poc_obrazu[vl_index2]
->poradi)
{if (poc_obrazu[vl_index2]->druh==1)
{f_sprava->e_obr1->Text =ChangeFileExt
(ExtractFileName(poc_obrazu[vl_index2]->t_cesta), "");
f_sprava->e_popis_obr1->Text=poc_obrazu[vl_index2]->t_nazev;}
if (poc_obrazu[vl_index2]->druh==2)
{f_sprava->e_obr2->Text=ChangeFileExt
(ExtractFileName(poc_obrazu[vl_index2]->t_cesta), "");
f_sprava->e_popis_obr2->Text=poc_obrazu[vl_index2]->t_nazev;}
if (poc_obrazu[vl_index2]->druh==3)
{f_sprava->e_obr3->Text=ChangeFileExt
(ExtractFileName(poc_obrazu[vl_index2]->t_cesta), "");
f_sprava->e_popis_obr3->Text=poc_obrazu[vl_index2]->t_nazev;}
if (poc_obrazu[vl_index2]->druh==4)
{f_sprava->e_vzorec1->Text=ChangeFileExt
(ExtractFileName(poc_obrazu[vl_index2]->t_cesta), "");
f_sprava->e_popis_vzor1->Text=poc_obrazu[vl_index2]->t_nazev;}
if (poc_obrazu[vl_index2]->druh==5)
{f_sprava->e_vzorec2->Text=ChangeFileExt
(ExtractFileName(poc_obrazu[vl_index2]->t_cesta), "");
f_sprava->e_popis_vzor2->Text=poc_obrazu[vl_index2]->t_nazev;}
if (poc_obrazu[vl_index2]->druh==6)
{f_sprava->e_vzorec3->Text=ChangeFileExt
(ExtractFileName(poc_obrazu[vl_index2]->t_cesta), "");
f_sprava->e_popis_vzor3->Text=poc_obrazu[vl_index2]->t_nazev;}}}}

```

Tato umožňuje, aby uživatel viděl všechny parametry termínu při jeho správě.

Nejprve se vytvoří lokální proměnné j pro úpravu českého názvu struktury t_seznamy, cislo_indexu, které udává, s kterým termínem se právě pracuje a znakové proměnné typu AnsiString prib1 až prib5 pro práci s příbuznými tématy.

Dalším krok funkce je, že se do proměnných prib1 až prib5 nahrají názvy příbuzných témat. Program projde všechny termíny databáze a testuje, jestli jejich čísla vyhovují číslům příbuzných témat daného termínu. Pokud ano, napíše program název tohoto příbuzného termínu do jedné z pěti proměnných.

Poté funkce aktivuje nebo zakáže komponenty, které budou nebo naopak nebudou potřeba a zobrazí parametry, které se nemusí nijak upravovat.

Následuje rozdělení německého názvu ze struktury na člen a německý název. Toto slovní spojení má tvar „německý název, člen“. Protože tvar tohoto slovního spojení je pevný,

program testuje všechny písmena zobrazovaného slovního spojení tak dlouho, dokud nenarazí na čárku. Poté kurzor poskočí ještě o dvě místa a písmena, která zůstala před kurzorem, vymaže. Dále program zobrazí do příslušného políčka německý a český název. Protože program při přidávání termínů doplňuje za slovíčka tabulátory, aby byly slovíčka ve sloupečku, je třeba je při opravování odstranit. To funkce dělá tak, že testuje všechna písmena slovíčka, dokud nenarazí na tabulátor. Potom označí text před kurzorem a znovu jej zobrazí. Po zobrazení parametrů slovníku následuje zobrazení textu termínu a jeho příbuzných témat. Pro zobrazení přidaných obrázků je potřeba funkce, která z celé cesty k souboru ve tvaru „obrazky\název_obrázku.bmp“ odstraní příponu a cestu a ponechá pouze čistý název obrázku. K tomu je použita funkce `ExtractFileName` a `ChangeFileExt`.

8.3.12 Funkce `abeceda()` pro zobrazení řecké abecedy

Funkce zde není vyobrazena celá, ale pouze jako příklad pro písmeno „α“.

```
void abeceda(void)
{f_hlavni->e_hledej->Text="abcalpha";
 int pozice=0, zacatek=0, do_konce=0;

do_konce = f_hlavni->re_text->Text.Length() - zacatek;
pozice = f_hlavni->re_text->FindText(f_hlavni->e_hledej->Text, zacatek,
do_konce, TSearchTypes() << stMatchCase);
if (pozice != -1)
{f_hlavni->re_text->SetFocus();
 f_hlavni->re_text->SelStart = pozice;
 f_hlavni->re_text->SelLength = f_hlavni->e_hledej->Text.Length();
 for (int i = 0; i < Screen->Fonts->Count; i++)
 if (Screen->Fonts->Strings[i]=="Symbol")
 f_hlavni->re_text->SelAttributes->Name=Screen->Fonts->Strings[i];
 f_hlavni->re_text->SelText="α";}
```

Protože v editoru kódu nejdou zobrazovat řecká písmena a při psaní se místo nich objevují jen otazníky, je potřeba je zobrazit nějak jinak. K tomu slouží funkce `abeceda()`.

Funkce je založena na principu vyhledávání. Místo řeckého písmena se v editoru kódu napíše jeho celý název. Tento název se potom vyhledává. Pokud program najde takové slovo (v našem příkladu sloha) Nastaví font v textovém editoru na „Symbol“, který písmeno „α“ zobrazuje jako „α“, a označený text se přepíše na již zmiňované „α“. Touto funkcí se elegantně zbavíme problému s řeckou abecedou.

8.4 Ošetření funkce hlavních událostí programu

8.4.1 Události na úvodním dialogovém okně

Protože je úvodní dialogové okno velice jednoduché a nachází se zde pouze šest komponent, není zde mnoho událostí. Kliknutí na jednu z komponent způsobí otevření jedné z databází. Tato událost je ošetřena příkazem `show()`, kde parametrem je formulář, který chceme otevřít.

8.4.2 Události na dialogových formulářích jednotlivých databází

Protože mají všechny databáze stejnou strukturu, bude vysvětlení, opět jako v předchozích bodech, na databázi DSP.

8.4.2.1 Vytvoření formuláře

Při vytvoření formuláře se nastaví velikost a vlastnosti některých komponent podle velikosti okna formuláře. Nejdůležitější akcí této události je načtení dat databáze jak pevně daných programátorem pomocí funkcí `adding()`, `adding1()`, `adding2()`, tak i dat přidanych uživatelem pomocí funkce `nacti()`. Při vytvoření databáze také program pohlídá, jestli není databáze plná a pokud je, tak znemožní vyvolání dialogového okna pro přidávání záznamů. Databáze *Mathematik* a *Physik* pochopitelně funkci `nacti()` a hlídání „plnosti“ databáze neobsahují.

8.4.2.2 Ošetření kliknutí na tlačítko „Suchen“

```
void __fastcall Tf_hlavni::t_hledejClick(TObject *Sender)
{int pozice, zacatek, do_konce;
AnsiString pamet=e_hledej->Text;
f_hlavni->re_vlastni->Visible=false;
if (re_text->SelLength&&re_text->SelLength == e_hledej->Text.Length())
    zacatek = re_text->SelStart + re_text->SelLength;
else
    {zacatek = 0;
f_hlavni->re_text->Width=f_hlavni->Width-270;
f_hlavni->re_text->Height=f_hlavni->Height-75;
reset();
int i=0,j=0;
for (index1=0; index1<count1; index1++)
    {if (poc_seznamu[index1]->podobor>i)
    {f_hlavni->re_text->SelAttributes->Size=13;
f_hlavni->re_text->Lines->Add("\n"+poc_seznamu[index1]
->t_nadpis1+"\n");
i=poc_seznamu[index1]->podobor;}
if(index1>=127&&j==0)
    {j=1;
f_hlavni->re_text->SelAttributes->Size=13;
f_hlavni->re_text->Lines->Add("\n Eigenstrom\n");}
show1(index1);}
i=0; j=0;
for (index1=0; index1<count1; index1++)
    {if (poc_seznamu[index1]->podobor>i)
    {f_hlavni->re_text->SelAttributes->Size=13;
f_hlavni->re_text->Lines->Add("\n"+poc_seznamu[index1]
->t_nadpis2+"\n");
i=poc_seznamu[index1]->podobor;}
if(index1>=127&&j==0)
    {j=1;
f_hlavni->re_text->SelAttributes->Size=13;
f_hlavni->re_text->Lines->Add("\n Vlastni\n");}
show2(index1);}
f_hlavni->re_text->Lines->Add("\n");
for (index=0; index<count0;index++)
    {show(index);}
try{for (int i=0; i<5;0 i++)
    {abeceda();}
catch (EInvalidOperation &A){};
e_hledej->Text=pamet;}

do_konce = re_text->Text.Length() - zacatek;
```

```

f_hlavni->re_text->Lines->SaveToFile("data.txt");
re_text->Text=re_text->Text.UpperCase();
pozice = re_text->FindText(e_hledej->Text.UpperCase(), zacatek, do_konce,
TSearchTypes() << stMatchCase);
if (pozice != -1)
{
    f_hlavni->re_text->Lines->LoadFromFile("data.txt");
    re_text->SetFocus();
    re_text->SelStart = pozice;
    re_text->SelLength = e_hledej->Text.Length();
    int iy = re_text->CaretPos.y;
    int zy = SendMessage(re_text->Handle, EM_GETFIRSTVISIBLELINE, 0, 0);
    SendMessage(re_text->Handle, EM_LINESCROLL, 0, iy - zy-1);
}
else
{
    f_hlavni->re_text->Lines->LoadFromFile("data.txt");
    ShowMessage("Nicht gefunden!");
    re_text->SelLength=!re_text->SelLength;}}[4]

```

Nejprve se nadefinují lokální proměnné pozice pro určení pozice kurzoru, začátek pro určení počátku vyhledávání, do_konce, která určuje délku do konce textu a proměnná a, která označuje funkci konec vyhledávání.

Dále program testuje, jestli uživatel hledá nové úplně slovo nebo chce pouze najít další polohu slova, které už program našel. V prvním případě se musí zobrazit úplně celá databáze (slovníky i detaily termínů), musí se nastavit proměnná zacatek na hodnotu nula a zvětšit textový editor přes celý formulář a zavolat funkci pro zobrazení řecké abecedy. Ve druhém případě je pouze potřeba nastavit proměnnou zacatek na hodnotu těsně za posledně nalezeným slovem.

Samotné vyhledávání je prováděno tak, že se určí velikost proměnné do_konce jako délka celého textu mínus proměnná pozice. Text v textovém editoru se uloží do textového souboru. Všechn text v textovém editoru se převede na velká písmena a zahájí se vyhledávání funkcí FindText(). Uložení do textového editoru a následné načtení je proto, aby bylo možné vyhledávat slova nezávisle na velikosti písmen.

Pokud bylo něco nalezeno, načtou se termíny z textového souboru, nastaví se zaměření do textového editoru a označí se nalezený text. Dále je potřeba, aby uživatel viděl, co bylo nalezeno, tedy aby kurzor skočil na polohu nalezeného slova. To je uskutečněno pomocí proměnných iy a zy. Načtení z textového souboru je mnohem rychlejší, než nové načítání databáze funkcí show(). Pokud nebylo nic nalezeno, Program načte data z textového souboru, informuje uživatele, že se nic nenašlo a řekne programu, že příští hledání musí jít od začátku.

8.4.2.3 Ošetření změny stromové struktury

Tato změna se uplatňuje při změně souřadnic x, y, z

```

void __fastcall Tf_hlavni::t_seznamChange(TObject *Sender, TTreeNode *Node)
{
    f_hlavni->t_zpet->Enabled=True;
    x = Node->ImageIndex;
    y = Node->SelectedIndex;
    z = Node->StateIndex;
    reset();
    counter1++;
    if (counter1>200)
    {
        for (int i=0; i<101; i++)
        {
            x1[i]=x1[i+100];
            y1[i]=y1[i+100];
            z1[i]=z1[i+100];
        }
        counter1=100;
        counter2=100;
    }
}

```

```

x1[counter1]=x;
y1[counter1]=y;
z1[counter1]=z;
citac1=1;
citac2=1;}

```

Protože už uživatel začal pracovat s programem, povolí se tlačítko zpět. Dále se nastaví souřadnice pro zobrazení termínu a provede funkce `rest`, která vymaže obsah komponent.

Následuje ošetření stavu, kdy je plná paměť pro tlačítka vpřed a zpět. Program přepíše všechny souřadnice na poloze od 100 do 200 na pozici 0 až 100, předchozí záznamy se vymažou.

Aktuální souřadnice se zapíší do paměti a čítače pro počítání počtu zmáčknutí tlačítek vpřed a zpět se nastaví na jedničku.

8.4.2.4 Ošetření kliknutí na tlačítko „Zurück“

```

void __fastcall Tf_hlavni::t_zpetClick(TObject *Sender)
{reset();
citac2=1;
x=x1[counter1-citac1];
y=y1[counter1-citac1];
z=z1[counter1-citac1];
zobraz();
counter2=(counter1-citac1);
citac1++;
f_hlavni->t_vpřed->Enabled=True;
if (counter1+1-citac1==0)
{f_hlavni->t_zpet->Enabled=false;}}

```

Nejprve se provede funkce `reset()`, která vymaže obsah komponent. Dále se nastaví čítač pro tlačítko vpřed na jedničku. Do souřadnic `x`, `y`, `z` se načtou souřadnice z paměti. Tyto souřadnice se vyvolávají způsobem, že se od aktuální hodnoty proměnné `counter1`, což je vlastně hodnota posledního záznamu v paměti odečte počet zmáčknutí tlačítka zpět. Následuje volání funkce `zobraz()`. `Counter2` počítá počet zmáčknutí tlačítka zpět a tím pádem i kolikrát lze zmáčknout tlačítko vpřed. Pokud je `counter1` na poslední hodnotě paměti, tlačítko zpět se stane neaktivní.

8.4.2.5 Ošetření kliknutí na tlačítko „Vorwärts“

```

void __fastcall Tf_hlavni::t_vpředClick(TObject *Sender)
{reset();
citac1=1;
x=x1[counter2+citac2];
y=y1[counter2+citac2];
z=z1[counter2+citac2];
zobraz();
citac2++;
f_hlavni->t_zpet->Enabled=true;
if (counter2+citac2>counter1)
{f_hlavni->t_vpřed->Enabled=false;}}

```

Nejprve se provede funkce `reset()`, která vymaže obsah komponent. Dále se nastaví čítač pro tlačítko zpět na jedničku. Do souřadnic `x`, `y`, `z` se načtou souřadnice z paměti. Tyto souřadnice se vyvolávají způsobem, že se od aktuální hodnoty proměnné `counter2`, což je vlastně hodnota posledního záznamu v paměti odečte počet zmáčknutí tlačítka vpřed.

Následuje volání funkce `zobraz()`. Pokud je `counter2` na poslední hodnotě paměti pro tlačítko vpřed, tlačítko vpřed se stane neaktivní.

8.4.2.5 Ošetření události změny velikosti formuláře

```
void __fastcall Tf_hlavni::FormResize(TObject *Sender)
{
    f_hlavni->b_podtrzitko->Width=f_hlavni->Width;
    f_hlavni->b_plocha->Height=f_hlavni->Height-80;
    f_hlavni->t_seznam->Height=f_hlavni->Height-210;
    zobraz();
}
```

Nejprve se změní velikost komponent pro lepší vzhled aplikace jako jsou plochy pro zvýraznění a podtrhovací čáry. Poté se zavolá funkce `zobraz`, která znovu zobrazí zobrazený záznam, aby nebyl nijak rozhozený nebo posunutý.

8.4.2.6 Ošetření funkce změny ComboBoxu pro příbuzná témata

```
void __fastcall Tf_hlavni::c_pribuzneChange(TObject *Sender)
{
    reset();
    for (index=0; index<count0; index++)
        {AnsiString temp = poc_terminu[index]->t_nadpis;
        if (temp==f_hlavni->c_pribuzne->Text)
            {x=poc_terminu[index]->obor;
            y=poc_terminu[index]->podobor;
            z=poc_terminu[index]->poradi;
            counter1++;
            if (counter1>200)
                {for (int i=0; i<101; i++)
                    {x1[i]=x1[i+100];
                    y1[i]=y1[i+100];
                    z1[i]=z1[i+100];}
                counter1=100;
                counter2=100;}
            x1[counter1]=x;
            y1[counter1]=y;
            z1[counter1]=z;
            citac1=1;
            citac2=1;}}
    zobraz();
}
```

Nejprve se zavolá funkce `reset()`, která vymaže obsah důležitých komponent.

Následně se pro všechny termíny do proměnné `temp` vyzkouší, zda se zvolený příbuzný termín rovná nějakému termínu z databáze. Po tom, co se termín najde, se do proměnných `x`, `y`, `z` nahrají souřadnice tohoto termínu. Se tyto souřadnice nahrají do paměti pro tlačítka vpřed a zpět a termín se zobrazí. Opět je zde ošetřen stav, kdy bude paměť pro tlačítka vpřed a zpět plná.

8.4.2.7 Ošetření události změny editoru pro hledání

```
void __fastcall Tf_hlavni::e_hledejChange(TObject *Sender)
{re_text->SelLength=0;}
```

Pokud se změní obsah editu pro hledání, znamená to, že uživatel změnil parametr pro hledání a program musí začít hledat od začátku.

8.4.2.8 Funkce pro ošetření přidání poznámky k termínu

```
void __fastcall Tf_hlavni::re_vlastniExit(TObject *Sender)
{f_hlavni->re_vlastni->Lines->SaveToFile(nazev_souboru+pripna);}
```

Funkce reaguje na vystoupení z editoru pro poznámku (nemá focus). Pokud editor pro poznámku focus ztratí, uloží se obsah této poznámky do textového editoru, ze kterého je potom při další volbě daného termínu načten.

8.4.2.9 Ošetření události při ukončení aplikace

```
void __fastcall Tf_hlavni::FormClose(TObject *Sender, TCloseAction &Action)
{uloz();}
```

Při ukončení aplikace je potřeba zavolat funkci uloz(), aby nebyly ztraceny termíny, které uživatel přidal. Tato událost je ošetřena pouze u databáze DSP.

8.4.2.10 Ošetření události kliknutí na stromovou strukturu

```
void __fastcall Tf_hlavni::t_seznamClick(TObject *Sender)
{zobraz();}
```

Při kliknutí na nějakou položku ve stromové struktuře se zavolá funkce zobraz().

8.4.2.11 Ošetření události při volbě „Drucken“ v hlavním menu

```
void __fastcall Tf_hlavni::Tisk1Click(TObject *Sender)
{if (PrintDialog1->Execute())
{if ((x!=4&&y==0)|| (x==4&&z==0)||x==1||x==2)
{re_text->Print(f_hlavni->Caption);}
else
{f_hlavni->re_text->Lines->SaveToFile("tisk.txt");
f_tisk->re_tisk->Clear();
f_tisk->re_tisk->Lines->LoadFromFile("tisk.txt");
f_tisk->re_tisk->Lines->Add(f_hlavni->re_vlastni->Text);
f_tisk->re_tisk->Print(f_tisk->Caption);
if (bmpName[0]!="0"||bmpName[3]!="0")
{TPrinter *Prntr = Printer();
Prntr->BeginDoc();
auto_ptr<Graphics::TBitmap>bmp(new Graphics::TBitmap());
TRect r(100, 100, 1500, 800);
for(int i = 0; i < 6; i++)
{try{bmp->LoadFromFile(bmpName[i]);}nacti ho ze souboru
if (r.Top>3000)
{Printer()->NewPage();
r.Top=100; r.Bottom=1000;}
Prntr->Canvas->StretchDraw(r,bmp.get());
Prntr->Canvas->TextOut(r.Right-500, r.Top-100 ,bmpCaption[i]);
r.Top+=1000;
r.Bottom+=1000;}
catch (EOpenError &e){;}}
Prntr->EndDoc();}}}
```

Funkce začíná ověřením podmínky, že pokud se správně uzavřel dialog pro tisk, tak se neprve zkontroluje a co se vůbec má tisknout. Pokud se tiskne více termínů najednou, je situace jednoduchá, protože stačí pouze vytisknout textový editor. Pokud ne, program načte data z textového editoru do textového souboru (tím se zabezpečí zachování formátu textu) a

nahraje jeho obsah do textového editoru formuláře pro tisk. Do tohoto formuláře se dále přidá text z poznámkového editoru a celý formulář pro tisk se vytiskne. Pokud se u tisknutého termínu vyskytují obrázky, program musí zavést novou třídu novou proměnnou Printer typu TPrinter. Tato třída je vlastně nové plátno pro tisk. Zde se vytvoří nová proměnná bmp typu Bitmap. Do které se nahrají všechny obrázky tisknutého termínu společně s jejich popisky. Funkce TRect slouží pro nastavení rozměrů plátna pro tisk. Dále je ošetřeno, že pokud je kurzor na plátně na poloze větší, než 3000 (s určitou rezervou se jedná o konec stránky), je generována nová stránka.

8.4.2.12 Ošetření události při volbě „Ende“ v hlavním menu

```
void __fastcall Tf_hlavni::Konec1Click(TObject *Sender)
{ Application->Terminate(); }
```

Aplikace se celá ukončí.

8.4.2.13 Ošetření události při volbě „Über den Program“ v hlavním menu

```
void __fastcall Tf_hlavni::OprogramulClick(TObject *Sender)
{ Application->MessageBox("Author: Pavel Škurek", "Über dem Program", MB_OK);
}
```

Program informační hlášku o autorovi.

8.4.2.14 Ošetření události při volbě „Zugeben einen Terminus“ v hlavním menu

```
void __fastcall Tf_hlavni::Pridat1Click(TObject *Sender)
{ f_password->Show(); }
```

Program vyvolá dialogové okno pro zadání hesla. Tato volba je pouze u databáze DSP.

8.4.2.15 Ošetření události při volbě „Bearbeiten einen Terminus“ v hlavním menu

```
void __fastcall Tf_hlavni::Sprava1Click(TObject *Sender)
{ f_password2->Show(); }
```

Program vyvolá dialogové okno pro zadání hesla. Tato volba je pouze u databáze DSP.

8.4.2.16 Ošetření události při volbě položky „Datei“ v hlavním menu

```
void __fastcall Tf_hlavni::S1Click(TObject *Sender)
{ if (vl_index!=127)
  f_hlavni->Sprava1->Enabled=true; }
```

Pokud je proměnná vl_index větší různý od 127, tzn. jsou přidány vlastní termíny, povol položku pro správu termínů.

8.4.2.16 Ošetření události při volbě „Links“ v hlavním menu

```
void __fastcall Tf_hlavni::Odkaz1Click(TObject *Sender)
{ f_odkaz->Show(); }
```

Program zobrazí formulář s odkazy na různé česko-německé slovníky.

8.4.2.17 Ošetření události při volbě „Hilfe“ v hlavním menu

```
void __fastcall Tf_hlavni::NpovdalClick(TObject *Sender)
{Application->HelpContext(1);}
```

Program zobrazí soubor s nápovědou.

8.4.2.18 Ošetření události při volbě „Andere Deteibase“ v hlavním menu

```
void __fastcall Tf_hlavni::AndereDateibaseClick(TObject *Sender)
{f_vyber->Show();
f_hlavni->Close();}
```

Program nejprve zobrazí úvodní dialogové okno a následně ukončí aktuální databázi.

Další funkce ošetřují události způsobené změnou na formuláři pro přidávání nebo správu termínů. Proto tyto funkce najdeme pouze v databázi DSP.

8.4.2.19 Funkce ošetřující kliknutí na tlačítko „Přidat“ („Zugeben“) a změnu editačních polí na formuláři pro přidání termínu

Funkce se volají tak, že když uživatel klikne na tlačítko „Přidat“ („Zugeben“) nebo změní jedno z editačních polí. Program virtuálně klikne na tlačítko b_pridej na formuláři pro prohlížení termínů a nastaví svůj „ukazatel“ Sender na komponentu, kterou tato událost vyvolala. Podle toho se program rozhodne, co má udělat. Tato metoda značně šetří použití nadbytečných konstant a příkazů.

```
void __fastcall Tf_hlavni::b_pridejClick(TObject *Sender)
{if (Sender==f_pridej->e_podobor||Sender==f_pridej->e_poradi)
{f_pridej->l_chyba->Caption="";
f_pridej->l_chyba2->Caption="";
for (int index=0; index<count0;index++)
{if (f_pridej->e_podobor->Enabled==true&&poc_terminu[index]
->obor==f_pridej->c_obor->ItemIndex+3&&poc_terminu[index]->podobor==
StrToInt(f_pridej->e_podobor->Text)&&poc_terminu[index]->poradi==1)
{f_pridej->l_chyba->Caption="!!!";}
if (f_pridej->e_podobor->Enabled==false&&poc_terminu[index]
->obor==f_pridej->c_obor->ItemIndex+3&&poc_terminu[index]
->poradi==StrToInt(f_pridej->e_poradi->Text)&&poc_terminu[index]
->podobor==f_pridej->c_podobor->ItemIndex+1)
{f_pridej->l_chyba2->Caption="!!!";}}}
```

První část funkce ošetřuje událost, která hlídá přiřazování čísel oborů nebo podoborů přidávaným termínům.

Pokud je Sender jedno z editačních polí, nejde o přidávání, ale pouze o zadání komponent. U obou komponent se vymažou Labely, které signalizují chybu. Potom se testuje podmínka, která testuje, jestli zadané číslo oboru nebo podoboru již v databázi nefiguruje. Pokud ano, nastaví se text v příslušném Labelu na „!!!“, což značí chybu.

```
else
{if (f_pridej->l_chyba->Caption=="!!!"||f_pridej->l_chyba2
->Caption=="!!!")
{Application->MessageBox("Falsche Zahl des Teilbereich oder der
Rangliste", "Error", MB_OK);}
```

Druhá část ošetřuje událost, která hlídá kliknutí na tlačítko „Přidat“ („Zugeben“). Než se začne cokoliv provádět, program zkontroluje, jestli jeden z Labelů nehlásí chybu. Pokud ano, objeví se chybové hlášení. Pokud ne pokračuje se příkazy pod tímto odstavcem.

```

else
    {if (f_pridej->e_obr1->Text=="")
        f_pridej->e_obr1->Text=0;
        if (f_pridej->e_obr2->Text=="")
            f_pridej->e_obr2->Text=0;
        if (f_pridej->e_obr3->Text=="")
            f_pridej->e_obr3->Text=0;
        if (f_pridej->e_popis_obr1->Text=="")
            f_pridej->e_popis_obr1->Text=0;
        if (f_pridej->e_popis_obr2->Text=="")
            f_pridej->e_popis_obr2->Text=0;
        if (f_pridej->e_popis_obr3->Text=="")
            f_pridej->e_popis_obr3->Text=0;
        if (f_pridej->e_vzorec1->Text=="")
            f_pridej->e_vzorec1->Text=0;
        if (f_pridej->e_vzorec2->Text=="")
            f_pridej->e_vzorec2->Text=0;
        if (f_pridej->e_vzorec3->Text=="")
            f_pridej->e_vzorec3->Text=0;
        if (f_pridej->e_popis_vzor1->Text=="")
            f_pridej->e_popis_vzor1->Text=0;
        if (f_pridej->e_popis_vzor2->Text=="")
            f_pridej->e_popis_vzor2->Text=0;
        if (f_pridej->e_popis_vzor3->Text=="")
            f_pridej->e_popis_vzor3->Text=0;

        {ShortString temp =f_pridej->e_poradi->Text;
        ShortString temp2=f_pridej->e_podobor->Text;
        ShortString temp3;
        int temp4;
        int prib1=0, prib2=0, prib3=0, prib4=0, prib5=0;
        AnsiString tab,tab2;
        if (f_pridej->e_nem_nazev->Text.Length()<=8)
            {tab=" \t\t\t\t\t";}
        if (f_pridej->e_nem_nazev->Text.Length()>8&&f_pridej->e_nem_nazev
            ->Text.Length()<=14)
            {tab=" \t\t\t\t\t";}
        if (f_pridej->e_nem_nazev->Text.Length()>14&&f_pridej->e_nem_nazev
            ->Text.Length()<=20)
            {tab=" \t\t\t\t\t";}
        if (f_pridej->e_nem_nazev->Text.Length()>20&&f_pridej->e_nem_nazev
            ->Text.Length()<=25)
            {tab=" \t\t\t\t\t";}
        if (f_pridej->e_nem_nazev->Text.Length()>25)
            {tab=" ";}
        if (f_pridej->e_ces_nazev->Text.Length()<=6)
            {tab2="\t\t\t\t\t\t\t";}
        if (f_pridej->e_ces_nazev->Text.Length()>6&&f_pridej->e_ces_nazev
            ->Text.Length()<=12)
            {tab2="\t\t\t\t\t\t\t";}
        if (f_pridej->e_ces_nazev->Text.Length()>12&&f_pridej->e_ces_nazev
            ->Text.Length()<=18)
            {tab2="\t\t\t\t\t\t\t";}
        if (f_pridej->e_ces_nazev->Text.Length()>18&&f_pridej->e_ces_nazev
            ->Text.Length()<=24)
            {tab2="\t\t\t\t\t\t\t";}
        if (f_pridej->e_ces_nazev->Text.Length()>24&&f_pridej->e_ces_nazev

```

```

->Text.Length()<=30)
    {tab2="\t";}
if (f_pridej->e_ces_nazev->Text.Length()>30)
    {tab2="";}
for (int index=0; index<count0; index++)
{AnsiString pr_text1=poc_terminu[index]->t_nadpis;
if (pr_text1==f_pridej->e_pribuzne1->Text)
    prib1=poc_terminu[index]->cislo;
if (pr_text1==f_pridej->e_pribuzne2->Text)
    prib2=poc_terminu[index]->cislo;
if (pr_text1==f_pridej->e_pribuzne3->Text)
    prib3=poc_terminu[index]->cislo;
if (pr_text1==f_pridej->e_pribuzne4->Text)
    prib4=poc_terminu[index]->cislo;
if (pr_text1==f_pridej->e_pribuzne5->Text)
    prib5=poc_terminu[index]->cislo;}

if (f_pridej->c_obor->ItemIndex==0)
    {temp3="Antialiasing";
    temp4=1;}
if (f_pridej->c_obor->ItemIndex==1&&f_pridej->c_podobor->ItemIndex==0)
    {temp3="Vzorkování";
    temp4=2;}
if (f_pridej->c_obor->ItemIndex==1&&f_pridej->c_podobor->ItemIndex==1)
    {temp3="Kvantování";
    temp4=3;}
if (f_pridej->c_obor->ItemIndex==1&&f_pridej->c_podobor->ItemIndex==2)
    {temp3="Analogově digitální převodníky";
    temp4=4;}
if (f_pridej->c_obor->ItemIndex==2)
    {temp3="Digitálně analogové převodníky";
    temp4=5;}
if (f_pridej->c_obor->ItemIndex==3)
    {temp3="Spektrální analýza";
    temp4=6;}
if ((f_pridej->c_obor->ItemIndex==0)|| (f_pridej->c_obor
->ItemIndex==2)|| (f_pridej->c_obor->ItemIndex==3))
    {poc_terminu[vl_index]=adding(f_pridej->c_obor->ItemIndex+3,
    StrToInt(temp2),1, count0+1, prib1, prib2, prib3, prib4, prib5, f_pridej
->e_nem_nazev->Text, f_pridej->e_pr_text1->Text,f_pridej->e_pr_text2
->Text, f_pridej->e_pr_text3->Text);
    count0++;
    vl_index++;
    poc_seznamu[vl_index1]=adding1(1,temp4,f_pridej->c_obor->Text,temp3,
    f_pridej->e_nem_nazev->Text+", "+f_pridej->c_clen->Text+tab,f_pridej
->e_ces_nazev->Text+tab2);
    count1++;
    vl_index1++;
    if (f_pridej->e_obr1->Text!="0"&&f_pridej->e_popis_obr1->Text!="0")
        {poc_obrazu[vl_index2]=adding2(f_pridej->c_obor->ItemIndex+3,
        StrToInt(temp2),1,1,1,f_pridej->e_popis_obr1->Text, "obrazky\\"
        +f_pridej->e_obr1->Text+".bmp");
        count2++;
        vl_index2++;}
    else
        {poc_obrazu[vl_index2]=adding2(f_pridej->c_obor->ItemIndex+3,
        StrToInt(temp2),1,0,0,"0","0");
        count2++;
        vl_index2++;}
if (f_pridej->e_obr2->Text!="0"&&f_pridej->e_popis_obr2->Text!="0")
    {poc_obrazu[vl_index2]=adding2(f_pridej->c_obor->ItemIndex+3,

```

```

        StrToInt(temp2),1,2,2,f_pridej->e_popis_obr2->Text,"obrazky\\"
        +f_pridej->e_obr2->Text+".bmp");
        count2++;
        vl_index2++;}
    else
        {poc_obrazu[vl_index2]=adding2(f_pridej->c_obor->ItemIndex+3,
        StrToInt(temp2),1,0,0,"0","0");
        count2++;
        vl_index2++;}
    if (f_pridej->e_obr3->Text!="0"&&f_pridej->e_popis_obr3->Text!="0")
        {poc_obrazu[vl_index2]=adding2(f_pridej->c_obor->ItemIndex+3,
        StrToInt(temp2),1,3,3,f_pridej->e_popis_obr3->Text,"obrazky\\"
        +f_pridej->e_obr3->Text+".bmp");
        count2++;
        vl_index2++;}
    else
        {poc_obrazu[vl_index2]=adding2(f_pridej->c_obor->ItemIndex+3,
        StrToInt(temp2),1,0,0,"0","0");
        count2++;
        vl_index2++;}
    if (f_pridej->e_vzorec1->Text!="0"&&f_pridej->e_popis_vzor1->Text!="0")
        {poc_obrazu[vl_index2]=adding2(f_pridej->c_obor->ItemIndex+3,
        StrToInt(temp2),1,1,4,f_pridej->e_popis_vzor1->Text,"obrazky\\"
        +f_pridej->e_vzorec1->Text+".bmp");
        count2++;
        vl_index2++;}
    else
        {poc_obrazu[vl_index2]=adding2(f_pridej->c_obor->ItemIndex+3,
        StrToInt(temp2),1,0,0,"0","0");
        count2++;
        vl_index2++;}
    if (f_pridej->e_vzorec2->Text!="0"&&f_pridej->e_popis_vzor2->Text!="0")
        {poc_obrazu[vl_index2]=adding2(f_pridej->c_obor->ItemIndex+3,
        StrToInt(temp2),1,2,5,f_pridej->e_popis_vzor2->Text,"obrazky\\"
        +f_pridej->e_vzorec2->Text+".bmp");
        count2++;
        vl_index2++;}
    else
        {poc_obrazu[vl_index2]=adding2(f_pridej->c_obor->ItemIndex+3,
        StrToInt(temp2),1,0,0,"0","0");
        count2++;
        vl_index2++;}
    if (f_pridej->e_vzorec3->Text!="0"&&f_pridej->e_popis_vzor3->Text!="0")
        {poc_obrazu[vl_index2]=adding2(f_pridej->c_obor->ItemIndex+3,
        StrToInt(temp2),1,3,6,f_pridej->e_popis_vzor3->Text,"obrazky\\"
        +f_pridej->e_vzorec3->Text+".bmp");
        count2++;
        vl_index2++;}
    else
        {poc_obrazu[vl_index2]=adding2(f_pridej->c_obor->ItemIndex+3,
        StrToInt(temp2),1,0,0,"0","0");
        count2++;
        vl_index2++;}

for(int j=14;j<f_hlavni->t_seznam->Items->Count;j++)
    {if(AddNode(f_hlavni->t_seznam->Items->Item[j],f_pridej->c_obor
    ->Text,f_pridej->e_nem_nazev->Text))
        {if (f_hlavni->t_seznam->Items->Item[j]->ImageIndex==0)
            {f_hlavni->t_seznam->Items->Item[j]->ImageIndex=
            poc_terminu[vl_index-1]->obor;
            f_hlavni->t_seznam->Items->Item[j]->SelectedIndex=

```

```

poc_termínu[vl_index-1]->podobor;
f_hlavni->t_seznam->Items->Item[j]->StateIndex=
poc_termínu[vl_index-1]->poradi;}}
f_pridej->e_podobor->Text=IntToStr(StrToInt(temp2)+1);}}

```

Nejprve se provede oprava editů pro přidávání obrázků. Může se stát, že uživatel obrázek nepřidá, ale vymaže obsah komponent. Pro správný chod by v těchto komponentách měla být vždy „0“. To je zaručeno pomocí podmínek, které v případě potřeby „0“ doplní.

Dalším krokem je definování lokálních proměnných ShortString temp, ShortString temp2, ShortString temp3 a int temp4 pro převod čísel oborů, podoborů a pořadí na jejich název a nebo naopak na číslo. Int prib1, prib2, prib3, prib4 a prib5 pro práci s příbuznými tématy a AnsiString tab a tab2 pro nastavení tabulátorů ve slovnících.

Následuje krok nastavení tabulátorů tab1 pro německé slovo a tab2 pro české slovo. Program vždy zkontroluje délku slova a podle jeho délky k němu přiřadí potřebný počet tabulátorů.

Pro přidání příbuzných témat je použita podmínka, která kontroluje názvy termínů v databázi. Pokud se termín v databázi shoduje s termínem z některého editačního pole, přiřadí program do příslušné proměnné prib1 až prib5 proměnnou číslo daného termínu. Pokud ne, zůstane v proměnných prib1 až prib5 nula. Proměnné prib1 až prib5 jsou potom přidány k termínu.

Další podmínky nastavují konstanty temp3 pro český název ve slovníku a temp4 pro hodnotu podoboru ve slovníku. Program podle hodnoty v ComboBoxu pro obor resp. ComboBoxu pro podobor přiřadí do těchto konstant potřebná čísla pro správnou funkci programu.

Poslední částí zdrojového kódu funkce je samotné přidání termínu a jeho zobrazení ve stromové struktuře. Nejdříve přichází na řadu podmínka, jedná-li se pouze o „jednoduchý“ termín s oborem a podoborem nebo „složitější“ termín s oborem, podoborem i pořadím. Syntaxe přidání „složitějšího“ termínu zde není uvedena z důvodu přehlednosti zdrojového kódu. Po kladném vyhodnocení podmínky se nejprve zavolá funkce adding(), která přidá záznam do struktury t_termin. Následně se zvětší hodnota proměnných count0 a vl_index o jedničku. Druhým krokem je zavolání funkce adding1(), která přidá záznam do struktury t_seznamy a opět se zvětší hodnota count1 a vl_index1 o jedničku. Přidání obrázku do struktury je o něco složitější. Program nejprve ověří, jestli je v obou sobě náležících editačních polích pro název obrázku a jeho popisek něco napsáno. Poté se teprve zavolá funkce adding2(). Funkce adding2() se zavolá i v případě, že podmínka není splněna, ale do struktury t_obraz se přidá obrázek, který bude mít všechny parametry „0“, tím pádem nikdy nebude vyhovovat žádné podmínce a nikdy nebude programem použit. Tím se zaručí správné načítání obrázků při novém spuštění aplikace. Po tomto kroku se zvětší hodnota proměnných count2 a vl_index2. Přidání záznamů do stromové struktury je uskutečňováno pomocí funkce AddNode. Tato funkce je popsána výše. Podmínka pod voláním této funkce přiřazuje souřadnice přidaným záznamům souřadnice. Při přidání má každý záznam hodnotu ItemIndex rovnu 0. To testuje podmínka, tento záznam najde a přiřadí mu souřadnice přidávaného termínu. Tímto je termín do databáze přidán a zbývá pouze automaticky zvýšit hodnotu v editačním poli pro podobor o jedničku, aby to nemusel provádět uživatel.

Další funkce na formuláři pro přidávání záznamů jsou pouze funkce na aktivování nebo deaktivování komponent, změnu jazyka a další různé ale ne složité a ne moc důležité drobnosti, takže si myslím, že je není třeba uvádět.

8.4.2.20 Ošetření funkce pro zobrazení upravovaného termínu při jeho správě

Funkce zobraz opět využívá ukazatel Sender. Přesněji, při změně posuvné lišty. Se vyše Sender na tlačítko b_zobraz na formuláři pro zobrazování termínů.

```
void __fastcall Tf_hlavni::b_zobrazClick(TObject *Sender)
{zobraz2();}
```

Při imaginárním stisknutí tlačítka b_zobraz se zavolá funkce zobraz2.

8.4.2.21 Funkce ošetřující kliknutí na tlačítko „Smazat“ („Ausradieren“) a změnu editačních polí na formuláři pro přidání termínu

Funkce zobraz opět využívá ukazatel Sender. Přesněji, při stisku tlačítka „Smazat“ („Ausradieren“). Se vyše Sender na tlačítko b_smaz na formuláři pro zobrazování termínů.

```
void __fastcall Tf_hlavni::b_smazClick(TObject *Sender)
{int cislo_indexu=f_sprava->sb_cislo_indexu->Position;
if (cislo_indexu!=126)
{for (int i=110; i<count2;i++)
{if (poc_termínu[cislo_indexu]->obor==poc_obrazu[i]
->obor&& poc_termínu[cislo_indexu]->podobor==poc_obrazu[i]->podobor
&& poc_termínu[cislo_indexu]->poradi==poc_obrazu[i]->poradi)
{delete (poc_obrazu[i]);
for (vl_index2=i; vl_index2<count2; vl_index2++)
{poc_obrazu[vl_index2]=poc_obrazu[vl_index2+1];}
vl_index2--;
count2--;
i--;}}
for (int j=0; j<f_hlavni->t_seznam->Items->Count; j++)
{if (poc_termínu[cislo_indexu]->obor==f_hlavni->t_seznam->Items
->Item[j]->ImageIndex&& poc_termínu[cislo_indexu]->podobor==f_hlavni
->t_seznam->Items->Item[j]->SelectedIndex&& poc_termínu[cislo_indexu]
->poradi==f_hlavni->t_seznam->Items->Item[j]->StateIndex)

{f_hlavni->t_seznam->Items->Item[j]->Delete();}}
delete (poc_termínu[cislo_indexu]);
delete (poc_seznamu[cislo_indexu]);
for (int vl_index=cislo_indexu; vl_index<count0; vl_index++)
{poc_termínu[vl_index]=poc_termínu[vl_index+1];
poc_seznamu[vl_index]=poc_seznamu[vl_index+1]; }
count0--;
vl_index--;
count1--;
vl_index1--;}
try{f_sprava->sb_cislo_indexu->Max--;
f_sprava->sb_cislo_indexu->Position=f_sprava->sb_cislo_indexu
->Position-1;
f_sprava->sb_cislo_indexuChange(Sender);}
catch (EInvalidOperation &E){f_sprava->Close();
f_hlavni->Sprava1->Enabled=false;}
f_hlavni->Pridat1->Enabled=true;}
```

Proměnná cislo_indexu udává mazaný záznam. Nejprve se provede podmínka, jestli je vůbec v databázi nějaký záznam, který lze smazat. Pokud ano, následuje další podmínka, cyklus s podmínkou, který vyhledá z databáze, které k termínu patří a smaže je. Všechny obrázky v paměti za smazaným se posunou o jedno místo dopředu. Po smazání obrázku se

logicky sníží velikost proměnných count2 a vl_index2 o jedničku. Dále je potřeba snížit i hodnotu pomocné proměnné i v cyklu, aby nedošlo k vynechání nějakého obrázku.

Další krok je vymazání záznamu ze stromové struktury. Program zkontroluje všechny záznamy ve stromové struktuře a vymaže ten, který má stejné souřadnice jako mazaný termín.

Po vymazání záznamu ze stromové struktury už se jen vymaže záznam ze struktury t_seznamy a t_termin. Všechny záznamy za mazaným se v paměti posunou o jedno místo dopředu a zmenší se velikost proměnných count0, vl_index, count1 a vl_index1.

Poslední krok funkce je zmenšení počtu záznamů na posuvné liště a zobrazení jiného záznamu. Program zobrazuje předchozí záznam pomocí Senderu, který virtuálně změní polohu posuvné lišty. Tento děj je ošetřen proti Run-Time chybě, která vznikne, pokud uživatel vymaže poslední záznam. Při této události se dialogové okno pro správu uzavře, a na formuláři pro zobrazování termínů se zakáže tento formulář znova otevírat. Poslední řádek funkce je zde pro to, že je databáze plná a již dříve bylo programem zakázáno otevírat dialogové okno pro přidávání záznamu.

8.4.2.22 Funkce ošetřující kliknutí na tlačítko „Opravit“ („Berichtigen“) a změnu editačních polí na formuláři pro přidání termínu

Funkce se volají tak, že když uživatel klikne na tlačítko „Opravit“ („Berichtigen“) nebo změní jedno z editačních polí. Program virtuálně klikne na tlačítko b_oprav na formuláři pro prohlížení termínů a nastaví svůj „ukazatel“ Sender na komponentu, kterou tato událost vyvolala. Podle toho se program rozhodne, co má udělat. Tato metoda značně šetří použití nadbytečných konstant a příkazů.

```
void __fastcall Tf_hlavni::b_zmenClick(TObject *Sender)
{if(Sender==f_sprava->e_podobor||Sender==f_sprava->e_poradi)
    {f_sprava->l_chyba->Caption="";
    f_sprava->l_chyba2->Caption="";
    for (int index=0; index<count0;index++)
        {if (f_sprava->e_podobor->Enabled==true&&poc_terminu[index]
            ->obor==f_sprava->c_obor->ItemIndex+3&&poc_terminu[index]
            ->podobor==StrToInt(f_sprava->e_podobor->Text)&&poc_terminu[index]
            ->poradi==1&&index!=f_sprava->sb_cislo_indexu->Position)
            {f_sprava->l_chyba->Caption="!!!";}
        if (f_sprava->e_podobor->Enabled==false&&poc_terminu[index]
            ->obor==f_sprava->c_obor->ItemIndex+3&&poc_terminu[index]
            ->poradi==StrToInt(f_sprava->e_poradi->Text)&&poc_terminu[index]
            ->podobor==f_sprava->c_podobor->ItemIndex+1&&index!=f_sprava
            ->sb_cislo_indexu->Position)
            {f_sprava->l_chyba2->Caption="!!!";}}}
```

První část funkce ošetřuje událost, která hlídá přiřazování čísel oborů nebo podoborů upravovaných termínům.

Pokud je Sender jedno z editačních polí, nejde o upravování, ale pouze o zadání komponent. U obou komponent se vymažou Labely, které signalizují chybu. Potom se testuje podmínka, která testuje, jestli zadané číslo oboru nebo podoboru již v databázi nefiguruje. Pokud ano, nastaví se text v příslušném Labelu na „!!!“, což značí chybu.

```
else
    {if ((f_sprava->l_chyba->Caption=="!!!"||f_sprava->l_chyba2
        ->Caption=="!!!")&&Sender==f_sprava->b_oprav)
        {Application->MessageBox("Falsche Zahl des Teilbereich oder der
        Rangliste","Error",MB_OK);}}
```

Druhá část ošetřuje událost, která hlídá kliknutí na tlačítko „Opravit“ („Berichtigen“). Než se začne cokoli provádět, program zkontroluje, jestli jeden z Labelů nehlásí chybu. Pokud ano, objeví se chybové hlášení. Pokud ne pokračuje se příkazy pod tímto odstavcem.

```

        if (f_sprava->e_obr1->Text=="")
            f_sprava->e_obr1->Text=0;
        if (f_sprava->e_obr2->Text=="")
            f_sprava->e_obr2->Text=0;
        if (f_sprava->e_obr3->Text=="")
            f_sprava->e_obr3->Text=0;
        if (f_sprava->e_popis_obr1->Text=="")
            f_sprava->e_popis_obr1->Text=0;
        if (f_sprava->e_popis_obr2->Text=="")
            f_sprava->e_popis_obr2->Text=0;
        if (f_sprava->e_popis_obr3->Text=="")
            f_sprava->e_popis_obr3->Text=0;
        if (f_sprava->e_vzorec1->Text=="")
            f_sprava->e_vzorec1->Text=0;
        if (f_sprava->e_vzorec2->Text=="")
            f_sprava->e_vzorec2->Text=0;
        if (f_sprava->e_vzorec3->Text=="")
            f_sprava->e_vzorec3->Text=0;
        if (f_sprava->e_popis_vzor1->Text=="")
            f_sprava->e_popis_vzor1->Text=0;
        if (f_sprava->e_popis_vzor2->Text=="")
            f_sprava->e_popis_vzor2->Text=0;
        if (f_sprava->e_popis_vzor3->Text=="")
            f_sprava->e_popis_vzor3->Text=0;
    if (Sender==f_sprava->b_oprav&&f_sprava->l_chyba->Caption!="!!!"&&f_sprava
->l_chyba2->Caption!="!!!")
    {int cislo_indexu=f_sprava->sb_cislo_indexu->Position;
    int index_obrazku;
    for (int i=110; i<count2;i++)
        {if (poc_terminu[cislo_indexu]->obor==poc_obrazu[i]
->obor&&f_sprava->poc_terminu[cislo_indexu]->podobor==poc_obrazu[i]
->podobor&&f_sprava->poc_terminu[cislo_indexu]->poradi==poc_obrazu[i]->poradi)
            {delete (poc_obrazu[i]);
            index_obrazku=i;}}
    for (int j=0;j<f_hlavni->t_seznam->Items->Count;j++)
        {if (poc_terminu[cislo_indexu]->obor==f_hlavni->t_seznam->Items->Item[j]
->ImageIndex&&poc_terminu[cislo_indexu]->podobor==f_hlavni->t_seznam
->Items->Item[j]->SelectedIndex&&poc_terminu[cislo_indexu]
->poradi==f_hlavni->t_seznam->Items->Item[j]->StateIndex)
            {f_hlavni->t_seznam->Items->Item[j]->Delete();}}
    delete (poc_terminu[cislo_indexu]);
    delete (poc_seznamu[cislo_indexu]);

    ShortString temp =f_sprava->e_poradi->Text;
    ShortString temp2=f_sprava->e_podobor->Text;
    ShortString temp3;
    int temp4;
    int prib1=0, prib2=0, prib3=0, prib4=0, prib5=0;
    AnsiString tab,tab2;
    if (f_sprava->e_nem_nazev->Text.Length()<=8)
        {tab=" \t\t\t\t\t";}
    if (f_sprava->e_nem_nazev->Text.Length()>8&&f_sprava->e_nem_nazev
->Text.Length()<=14)
        {tab=" \t\t\t\t\t";}
    if (f_sprava->e_nem_nazev->Text.Length()>14&&f_sprava->e_nem_nazev
->Text.Length()<=20)

```

```

        {tab=" \t\t";}
if (f_sprava->e_nem_nazev->Text.Length()>20&&f_sprava->e_nem_nazev
->Text.Length()<=26)
    {tab=" \t";}
if (f_sprava->e_nem_nazev->Text.Length()>26)
    {tab="";}
if (f_sprava->e_ces_nazev->Text.Length()<=6)
    {tab2="\t\t\t\t\t";}
if (f_sprava->e_ces_nazev->Text.Length()>6&&f_sprava->e_ces_nazev
->Text.Length()<=12)
    {tab2="\t\t\t\t\t";}
if (f_sprava->e_ces_nazev->Text.Length()>12&&f_sprava->e_ces_nazev
->Text.Length()<=18)
    {tab2="\t\t\t\t\t";}
if (f_sprava->e_ces_nazev->Text.Length()>18&&f_sprava->e_ces_nazev
->Text.Length()<=24)
    {tab2="\t\t\t\t\t";}
if (f_sprava->e_ces_nazev->Text.Length()>24&&f_sprava->e_ces_nazev
->Text.Length()<=30)
    {tab2="\t\t\t\t\t";}
if (f_sprava->e_ces_nazev->Text.Length()>30)
    {tab2="";}
for (int index=0; index<count0; index++)
    {AnsiString pr_text1=poc_terminu[index]->t_nadpis;
    if (pr_text1==f_sprava->e_pribuzne1->Text)
        prib1=poc_terminu[index]->cislo;
    if (pr_text1==f_sprava->e_pribuzne2->Text)
        prib2=poc_terminu[index]->cislo;
    if (pr_text1==f_sprava->e_pribuzne3->Text)
        prib3=poc_terminu[index]->cislo;
    if (pr_text1==f_sprava->e_pribuzne4->Text)
        prib4=poc_terminu[index]->cislo;
    if (pr_text1==f_sprava->e_pribuzne5->Text)
        prib5=poc_terminu[index]->cislo;}
if (f_sprava->c_obor->ItemIndex==0)
    {temp3="Antialiasing";
    temp4=1;}
if (f_sprava->c_obor->ItemIndex==1&&f_pridej->c_podobor->ItemIndex==0)
    {temp3="Vzorkování";
    temp4=2;}
if (f_sprava->c_obor->ItemIndex==1&&f_pridej->c_podobor->ItemIndex==1)
    {temp3="Kvantování";
    temp4=3;}
if (f_sprava->c_obor->ItemIndex==1&&f_pridej->c_podobor->ItemIndex==2)
    {temp3="Analogově digitální převodníky";
    temp4=4;}
if (f_sprava->c_obor->ItemIndex==2)
    {temp3="Digitálně analogové převodníky";
    temp4=5;}
if (f_sprava->c_obor->ItemIndex==3)
    {temp3="Spektrální analýza";
    temp4=6;}
if ((f_sprava->c_obor->ItemIndex==0)|| (f_sprava->c_obor->ItemIndex==2)||
(f_sprava->c_obor->ItemIndex==3))
    {poc_terminu[cislo_indexu]=adding(f_sprava->c_obor->ItemIndex+3,
StrToInt(temp2),1, count0+1, prib1, prib2, prib3, prib4, prib5,
f_sprava->e_nem_nazev->Text, f_sprava->e_pr_text1->Text,f_sprava
->e_pr_text2->Text, f_sprava->e_pr_text3->Text) ;
poc_seznamu[cislo_indexu]=adding1(1,temp4,f_sprava->c_obor->Text,temp3,
f_pridej->e_nem_nazev->Text+", "+f_pridej->c_clen->Text+tab,f_sprava
->e_ces_nazev->Text+tab2);

```

```

if (f_sprava->e_obr1->Text!="0"&&f_sprava->e_popis_obr1->Text!="0")
    {poc_obrazu[index_obrazku-5]=adding2(f_sprava->c_obor->ItemIndex+3,
    StrToInt(temp2),1,1,1,f_sprava->e_popis_obr1->Text,"obrazky\\"
    +f_sprava->e_obr1->Text+".bmp");}
else
    {poc_obrazu[index_obrazku-5]=adding2(f_sprava->c_obor->ItemIndex+3,
    StrToInt(temp2),1,0,0,"0","0");}
if (f_sprava->e_obr2->Text!="0"&&f_sprava->e_popis_obr2->Text!="0")
    {poc_obrazu[index_obrazku-4]=adding2(f_sprava->c_obor->ItemIndex+3,
    StrToInt(temp2),1,2,2,f_sprava->e_popis_obr2->Text,"obrazky\\"
    +f_sprava->e_obr2->Text+".bmp");}
else
    {poc_obrazu[index_obrazku-4]=adding2(f_sprava->c_obor->ItemIndex+3,
    StrToInt(temp2),1,0,0,"0","0");}
if (f_sprava->e_obr3->Text!="0"&&f_sprava->e_popis_obr3->Text!="0")
    {poc_obrazu[index_obrazku-3]=adding2(f_sprava->c_obor->ItemIndex+3,
    StrToInt(temp2),1,3,3,f_sprava->e_popis_obr3->Text,"obrazky\\"+
    f_sprava->e_obr3->Text+".bmp");}
else
    {poc_obrazu[index_obrazku-3]=adding2(f_sprava->c_obor->ItemIndex+3,
    StrToInt(temp2),1,0,0,"0","0");}
if (f_sprava->e_vzorec1->Text!="0"&&f_sprava->e_popis_vzor1->Text!="0")
    {poc_obrazu[index_obrazku-2]=adding2(f_sprava->c_obor->ItemIndex+3,
    StrToInt(temp2),1,1,4,f_sprava->e_popis_vzor1->Text,"obrazky\\"+
    f_sprava->e_vzorec1->Text+".bmp");}
else
    {poc_obrazu[index_obrazku-2]=adding2(f_sprava->c_obor->ItemIndex+3,
    StrToInt(temp2),1,0,0,"0","0");}
if (f_sprava->e_vzorec2->Text!="0"&&f_sprava->e_popis_vzor2->Text!="0")
    {poc_obrazu[index_obrazku-1]=adding2(f_sprava->c_obor->ItemIndex+3,
    StrToInt(temp2),1,2,5,f_sprava->e_popis_vzor2->Text,"obrazky\\"+
    f_sprava->e_vzorec2->Text+".bmp");}
else
    {poc_obrazu[index_obrazku-1]=adding2(f_sprava->c_obor->ItemIndex+3,
    StrToInt(temp2),1,0,0,"0","0");}
if (f_sprava->e_vzorec3->Text!="0"&&f_sprava->e_popis_vzor3->Text!="0")
    {poc_obrazu[index_obrazku]=adding2(f_sprava->c_obor->ItemIndex+3,
    StrToInt(temp2),1,3,6,f_sprava->e_popis_vzor3->Text,"obrazky\\"+
    f_sprava->e_vzorec3->Text+".bmp");}
else
    {poc_obrazu[index_obrazku]=adding2(f_sprava->c_obor->ItemIndex+3,
    StrToInt(temp2),1,0,0,"0","0");}
for(int j=14;j<f_hlavni->t_seznam->Items->Count;j++)
    {if(AddNode(f_hlavni->t_seznam->Items->Item[j],f_sprava->c_obor
    ->Text,f_sprava->e_nem_nazev->Text))
        {if (f_hlavni->t_seznam->Items->Item[j]->ImageIndex==0)
            {f_hlavni->t_seznam->Items->Item[j]->ImageIndex=
            poc_terminu[vl_index-1]->obor;
            f_hlavni->t_seznam->Items->Item[j]->SelectedIndex=
            poc_terminu[vl_index-1]->podobor;
            f_hlavni->t_seznam->Items->Item[j]->StateIndex=
            poc_terminu[vl_index-1]->poradi;}}}}

povol=0;
uloz();
res=1;}
else
    {if (f_sprava->r_jazyk->ItemIndex==0&&res==1)
        {Application->MessageBox("Pro správný chod se musí aplikace ukončit a
        pustit znova","Reset",MB_OK);

```

```

Application->Terminate();}
if (f_sprava->r_jazyk->ItemIndex==1&&res==1)
    {Application->MessageBox("Für guter Lauf muss man das Program sich
    beenden und dann noch einmal anlassen", "Reset", MB_OK);
    Application->Terminate();}}

```

Nejprve se provede oprava editů pro přidávání obrázků. Může se stát, že uživatel obrázek nepřidá, ale vymaže obsah komponent. Pro správný chod by v těchto komponentách měla být vždy „0“. To je zaručeno pomocí podmínek, které v případě potřeby „0“ doplní.

Proměnná `cislo_indexu` udává mazaný záznam. Nejprve se provede podmínka, jestli se jednalo o zmáčknutí tlačítka „Opravit“ („Berichtigen“), tím se zaručí to, že se do proměnné `res` uloží hodnota 1, která oznámí programu, že je potřeba ho restartovat. Pokud ano, následuje další podmínka, cyklus s podmínkou, který vyhledá z databáze, které k termínu patří a smaže je. Dále je potřeba nahrát hodnotu proměnné `i` do proměnné `cislo_indexu`, aby se vědělo, na kterou pozici si následný opravený termín znova nahraje. Další krok je vymazání záznamu ze stromové struktury. Program zkontroluje všechny záznamy ve stromové struktuře a vymaže ten, který má stejné souřadnice jako mazaný termín.

Po vymazání záznamu ze stromové struktury už se jen vymaže záznam ze struktury `t_seznamy` a `t_termín`.

Dalším krokem je definování lokálních proměnných `ShortString temp`, `ShortString temp2`, `ShortString temp3` a `int temp4` pro převod čísel oborů, podoborů a pořadí na jejich název a nebo naopak na číslo. `int prib1`, `prib2`, `prib3`, `prib4` a `prib5` pro práci s příbuznými tématy a `AnsiString tab` a `tab2` pro nastavení tabulátorů ve slovnících.

Následuje krok nastavení tabulátorů `tab1` pro německé slovo a `tab2` pro české slovo. Program vždy zkontroluje délku slova a podle jeho délky k němu přiřadí potřebný počet tabulátorů.

Pro přidání příbuzných témat je použita podmínka, která kontroluje názvy termínů v databázi. Pokud se termín v databázi shoduje s termínem z některého editačního pole, přiřadí program do příslušné proměnné `prib1` až `prib5` proměnnou `cislo` daného termínu. Pokud ne, zůstane v proměnných `prib1` až `prib5` nula. Proměnné `prib1` až `prib5` jsou potom přidány k termínu.

Další podmínky nastavují konstanty `temp3` pro český název ve slovníku a `temp4` pro hodnotu podoboru ve slovníku. Program podle hodnoty v `ComboBoxu` pro obor resp. `ComboBoxu` pro podobor přiřadí do těchto konstant potřebná čísla pro správnou funkci programu.

Poslední částí zdrojového kódu funkce je nové přidání termínu a jeho zobrazení ve stromové struktuře. Nejdříve přichází na řadu podmínka, jedná-li se pouze o „jednoduchý“ termín s oborem a podoborem nebo „složitější“ termín s oborem, podoborem i pořadím. Syntaxe přidání „složitějšího“ termínu zde není uvedena z důvodu přehlednosti zdrojového kódu. Po kladném vyhodnocení podmínky se nejprve zavolá funkce `adding()`, která přidá na pozici proměnné `cislo_indexu` v paměti záznam do struktury `t_termín`. Druhým krokem je zavolání funkce `adding1()`, která přidá záznam do struktury `t_seznamy` na pozici proměnné `cislo_indexu` v paměti. Přidání obrázku do struktury je o něco složitější. Program nejprve ověří, jestli je v obou sobě náležících editačních polích pro název obrázku a jeho popis něco napsáno. Poté se teprve zavolá funkce `adding2()`. Funkce `adding2()` se zavolá i v případě, že podmínka není splněna, ale do struktury `t_obraz` se přidá obrázek, který bude mít všechny parametry „0“, tím pádem nikdy nebude vyhovovat žádné podmínce a nikdy nebude programem použit. Tím se zaručí správně načítání obrázků při novém spuštění aplikace. První obrázek se uloží do paměti na pozici `cislo_indexu` a poslední na pozici `cislo_indexu+5`. Přidání záznamů do stromové struktury je uskutečňováno pomocí funkce `AddNode`. Tato funkce je popsána výše. Podmínka pod voláním této funkce přiřazuje souřadnice přidáním

záznamům souřadnice. Při přidání má každý záznam hodnotu ItemIndex rovnu 0. To testuje podmínka, tento záznam najde a přiřadí mu souřadnice přidávaného termínu. Tímto je termín do databáze přidán.

Protože je po upravení termínu funkce potřeba restartovat, zavolá se funkce uloz() a hodnota proměnné res pro nutnost resetu se nasataví na jednočku. Poslední podmínka rozhoduje o tom, jestli má být informační hláška o restart v českém nebo německém jazyce. Při provedení těchto podmínek se aplikace ukončí.

Další funkce na formuláři pro správu záznamů jsou pouze funkce na aktivování nebo deaktivování komponent, změnu jazyka a další různé ale ne složité a ne moc důležité drobnosti, takže si myslím, že je není třeba uvádět.

8.5 Nedostatky programu

První chybou programu je, že program neumí zobrazovat horní a dolní indexy v textu. Protože je použit jednoduchý textový editor. Změna na horní nebo dolní index by se v programu musela provádět kódem, který i tak změní pouze font upravovaného slova. Protože je výskyt horních a dolních indexů vyskytuje bez jakéhokoliv pravidla. Tento kód se nedá do funkce adding zakomponovat.

Druhá chyba programu, je chyba, kterou nedokážu vysvětlit, chyba se projevuje tak, že při prohlížení některého termínu může najednou vyskočit Run-Time chyba, která upozorňuje na špatný přístup do paměti. Tato chyba nijak neovlivní funkci programu a program pracuje bez problémů dál. Tato chyba se objevuje jen na některých počítačích a opravdu velmi zřídka. Bohužel tato chyba nejde odstranit pomocí funkce try, catch, protože při ladění programu se Run-Time chyba hlásí v souboru dsstring.h, kde nelze odstranit.

9 Závěr

Výsledkem mé práce je podle mého názoru vcelku jednoduchý a přehledný program, který zvládne ovládat snad každý uživatel. Jak už je mnohokrát uvedeno, výsledná práce se dá rozdělit na 3 databáze, Mathematik, Physik a DSP.

V prvních dvou databázích jsou vcelku podrobně vysvětleny všechny pojmy z matematiky a fyziky obsahující látku probíranou na střední škole. Snažil jsem se, aby obě databáze obsahovali opravdu všechny termíny a veškerou problematiku, která se v těchto odvětvích může objevit. To se mi doufám povedlo.

Ve třetí databázi jsem se snažil najít a německy vysvětlit co nejvíce pojmů z oblasti zpracování číslicového signálu. Protože zpracování číslicového signálu je široká a rychle se rozvíjející oblast, určitě jsem neuvedl všechny výrazy, snažil jsem se pouze uvést ty, které musí každý elektrotechnik bezpodmínečně znát a vědět, co znamenají. To, jestli jsem vybral správné, výrazy musí posoudit až praxe.

Samotná tvorba programu se samozřejmě neobešla bez menších a větších problémů. Většinu z nich se mi podařilo vyřešit a program tedy pracuje správně. Při ladění programu jsem se snažil objevit co nejvíce problémů, které mohou nastat při nesprávné manipulaci s programem. Snažil jsem se odstranit všechny Run-Time chyby, jako jsou například nesprávné operace s pamětí, načítání neexistujících souborů, dělení nulou atd. Jako tvůrce programu, který ví přesně, jak s programem zacházet je tato činnost velice složitá, protože vím, jak jsou jednotlivé funkce koncipovány. I tak si myslím, že jsem snad odstranil všechny chyby tohoto charakteru. Druhým typem možností vzniklých chyb je neznalost uživatele a zadávání špatných dat do programu. Tento problém jsem se snažil vyřešit tím, že program většinu parametrů kontroluje a pokud nějaký údaj nesedí, oznámí to uživateli nebo problém

sám opraví. Způsob, jak předejít zadání špatných dat, je ten, že program spoustu úkonů dělá automaticky. V kapitole 8.5 jsou drobné chyby, které se mi nepodařilo odstranit. Tyto chyby by možná zkušeného programátora nijak nepřekvapili a bez problémů by je odstranil. Moje omezené schopnosti na tento problém nestačily. I tak si ale myslím, že program funguje velice dobře a splňuje všechny požadavky, které na něj byly kladeny. Možnost odstranění vidím v použití jiné metody zobrazování nebo dokonce jiného programovacího jazyka.

10 Literatura

- [1] JAN, J.: Číslíková filtrace, analýza a restaurace signálu, VUTÍUM, 2002
- [2] ŠEBESTA, V. SMÉKAL, Z.: Signály a soustavy přednášky. VUTÍUM, 2004. 150 s.
- [3] GESCHEIDTOVÁ, E., REZ, J., STEINBAUER, M.: Měření v elektrotechnice. VUTÍUM, 2002. 150 s.
- [4] KADLEC, V.: Učíme se programovat v Borland C++ Builder a jazyce C++, COMPUTER PRESS, 2004
- [5] http://deutsches_alphabet.know-library.net/
- [6] http://www.elo-formel-datenbank.de/Elektrotechnik/Allgemeines/Griechisches_Alphabet.htm
- [7] http://www.alphabetpage.com/german/alphabet_for_germans/index.html
- [8] <http://www.wikipedia.de>
- [9] <http://www.wikipedia.cz>
- [10] <http://slovník.seznam.cz>
- [11] <http://slovník.seznam.cz>
- [12] <http://www.wikipedia.cz>
- [13] <http://www.wikipedia.de>
- [14] <http://hippo.feld.cvut.cz/bicak/los/cviceni-07/los-cv075.html>
- [15] <http://www.encyclopedia.com/sl/artic/n/ni/nihanje.html>
- [16] <http://www.fi.muni.cz/usr/staudek/rehak/diplomovka.html>
- [17] <http://www.fi.muni.cz/usr/jkucera/pv109/2003p/xblazek2.htm>
- [18] http://www.volny.cz/mills1/digital/digitalizace_obrazu/index.htm
- [19] <http://www.earchiv.cz/b02/b0610001.php3>
- [20] http://lucy.troja.mff.cuni.cz/~tichy/kap5/5_4_2.html
- [21] <http://gerstner.felk.cvut.cz/biolab/newbiolab/teach/prezentace/CengerovaCanda>
- [22] http://www.spse-ke.fei.tuke.sk/elektro/selektory/frekvenc/frek_sel.htm
- [23] <http://www.aligisassu.it/p/pc/pcm.html>
- [24] <http://www.csie.ntu.edu.tw>
- [25] <http://www.builder.cz>