Moscow State Technical University
IU-4 department: «The design and technology of the electronic equipment»

Menshov Kirill Anatolievitch

The graduation work

The laboratory complex for research of digital audiostreams compression algorithms

Research manager: PhD., professor Myslovsky E.V.

Moscow
2004

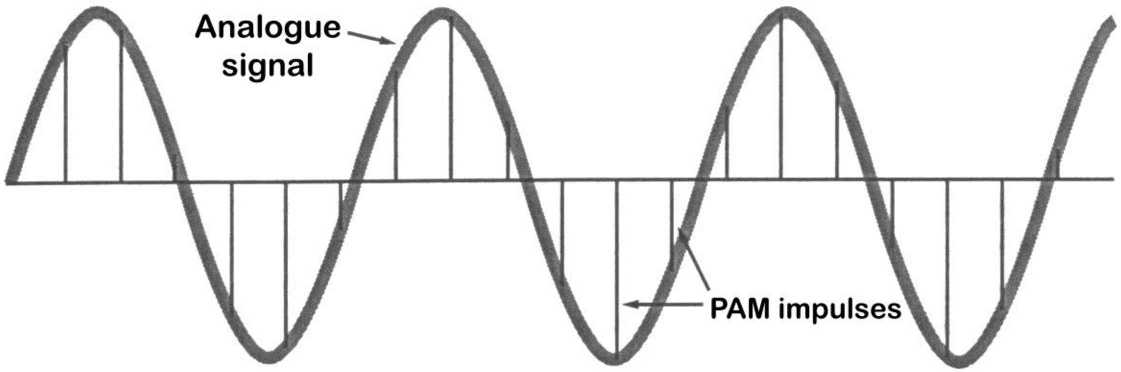
Goals and objectives

Goal: to develop a laboratory complex for creation, research and optimization of algorithms for digital audiostreams compression

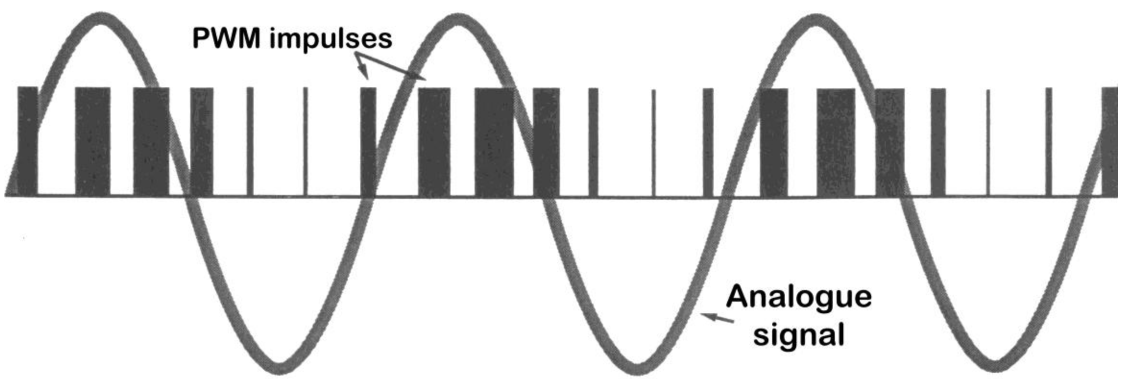
Objectives:

- Perform the analysis of methods and formats for a digital sound representation
- Investigate existing methods of digital audiostreams compression
- Perform the analysis of existing hardware for digital audiostreams processing and to choose ones for implementation of a complex
- Develop a methodology for creation of applications for digital audiostreams compression and working with the laboratory complex
- Implement an algorithm of audiostream compression in real time, providing significant reduction of data volume without essential loss of quality
- Offer a methods for quality testing of various algorithms for audiostreams compression

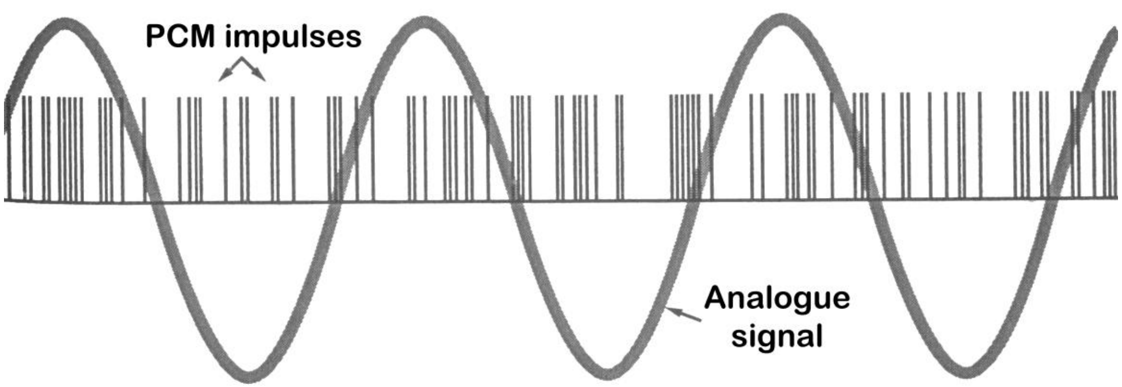
Representation of a sound in a digital form



Pulse-amplitude modulation



Pulse-width modulation



Pulse-code modulation

The format for storing the non compressed digital sound representation

Offset	***Name***	***Length***	***Description***
00h	rID	4h	Format ID: «RIFF»
04h	rLen	4h	The length of the data in the next chunk
08h	rData	rLen	

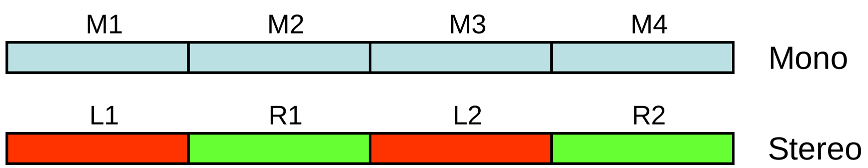
Offset	***Name***	***Length***	***Description***
00h	wID	4h	Chunk ID: «WAVE»
04h	Format	14h	
18h	waveData	—	

Offset	***Name***	***Length***	***Description***
00h	fID	4	Chunk ID: «fmt»
04h	fLen	4	Length of data in the Format chunk
08h	wFormatTag	2	
0Ah	nChannels	2	Number of channels
0Ch	nSamplesPerSec	2	Playback frequency
0Eh	nAvgBytesPerSec	2	The average number of bytes per sec., the data should be transferred at
10h	nBlockAlign	2	The block alignment of the data in the data chunk
12h	FormatSpecific	2	Format specific data area

wFormatTag value
WAVE_FORMAT_PCM (0x0001)
FORMAT_MULAW (0x0101)
IBM_FORMAT_ALAW (0x0102)
IBM_FORMAT_ADPCM (0x0103)

Offset	***Name***	***Length***	***Description***
00h	dID	4h	Chunk ID: «data»
04h	dLen	4h	Length of data in the dData field
08h	dData	dLen	The actual waveform data

Samples order in the dData block:



The audiodata compression methods

Nonlinear compression:

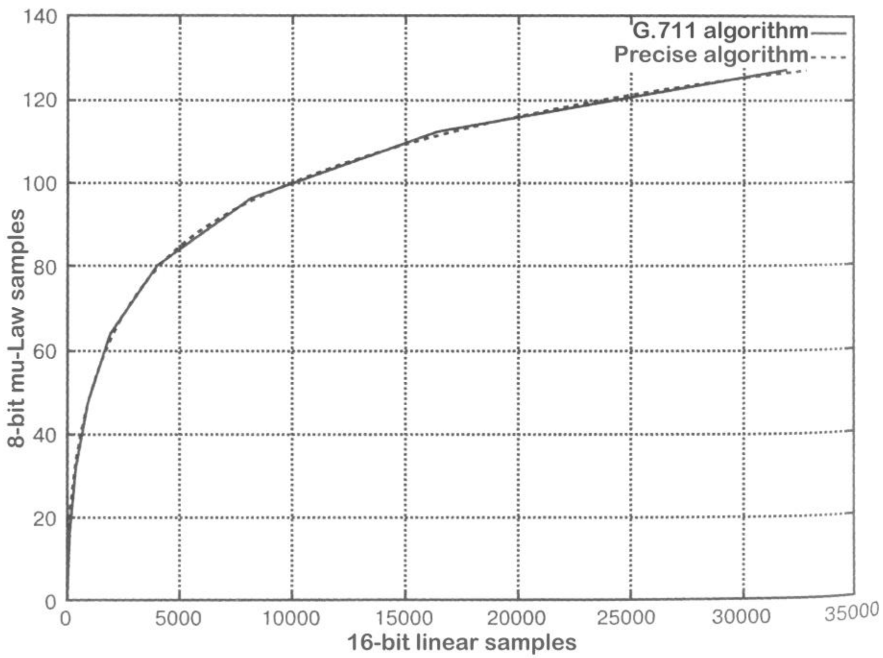
μ-Law: $sm = sign(s) \cdot \dfrac{\log(1+255|s|)}{\log(1+255)}$

Considering the positive values only, the formula becomes:

$$sm = \frac{\log(1+255s)}{\log(1+255)}$$

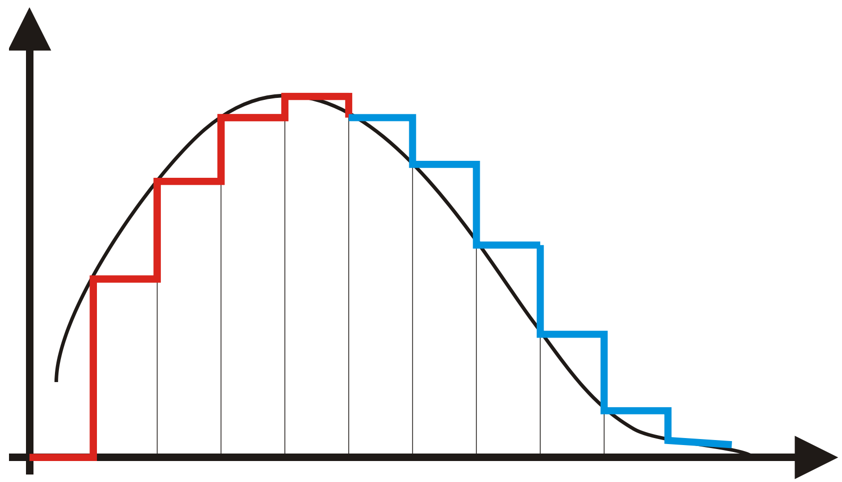
Reverse conversion:

$$s = \frac{256 \cdot sm - 1}{255}$$



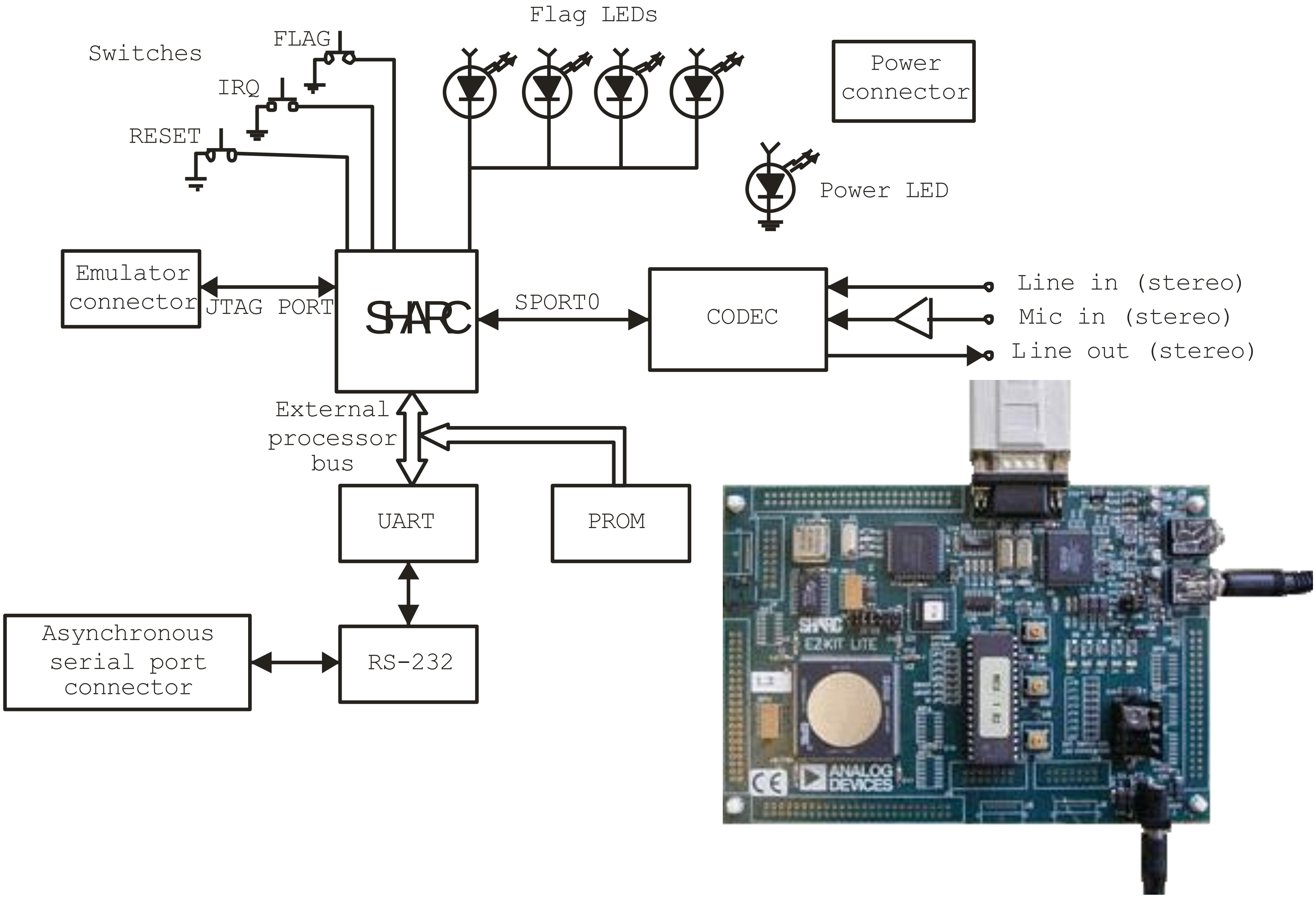
A-Law: $sA = \begin{cases} \dfrac{A}{1+\ln A}(s) & s \le \dfrac{1}{A} \\ \dfrac{1+\ln(As)}{1+\ln A} & \dfrac{1}{A} \le |s| \le 1 \end{cases}$

Differential coding:

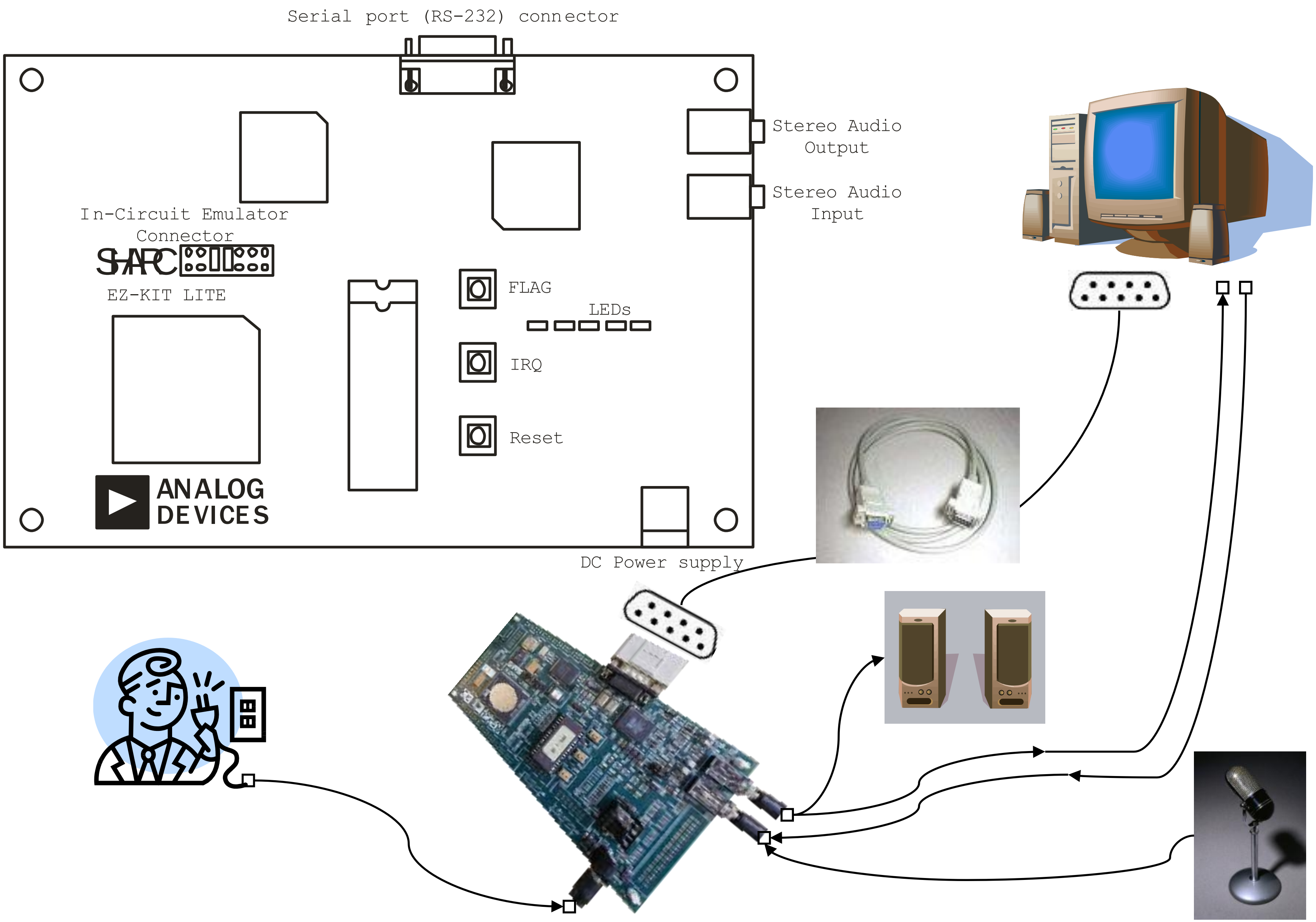


$$D_i = S_{i-1} - S_i$$

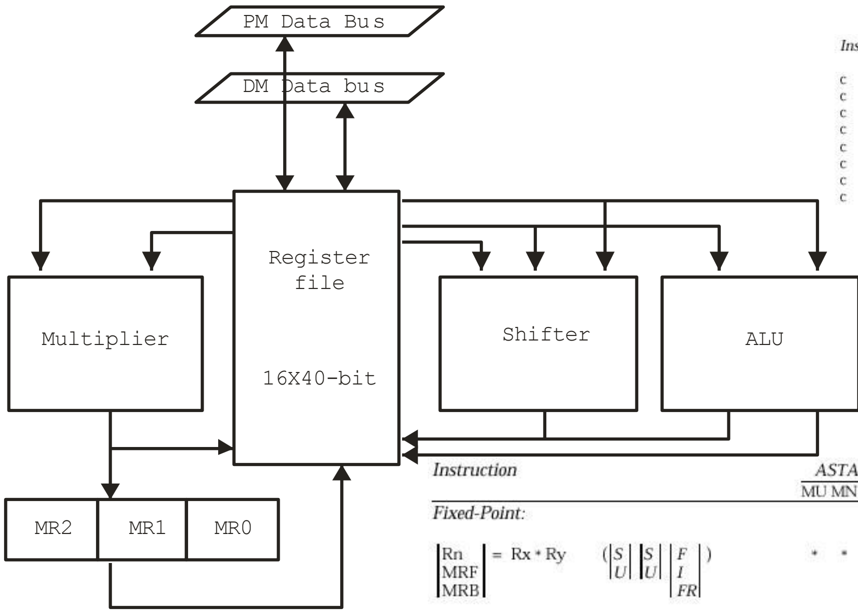
The SHARC EZ-KIT Lite evaluation module



External connections of the SHARC EZ-KIT Lite



Central processing unit of the SHARC EZ-KIT Lite

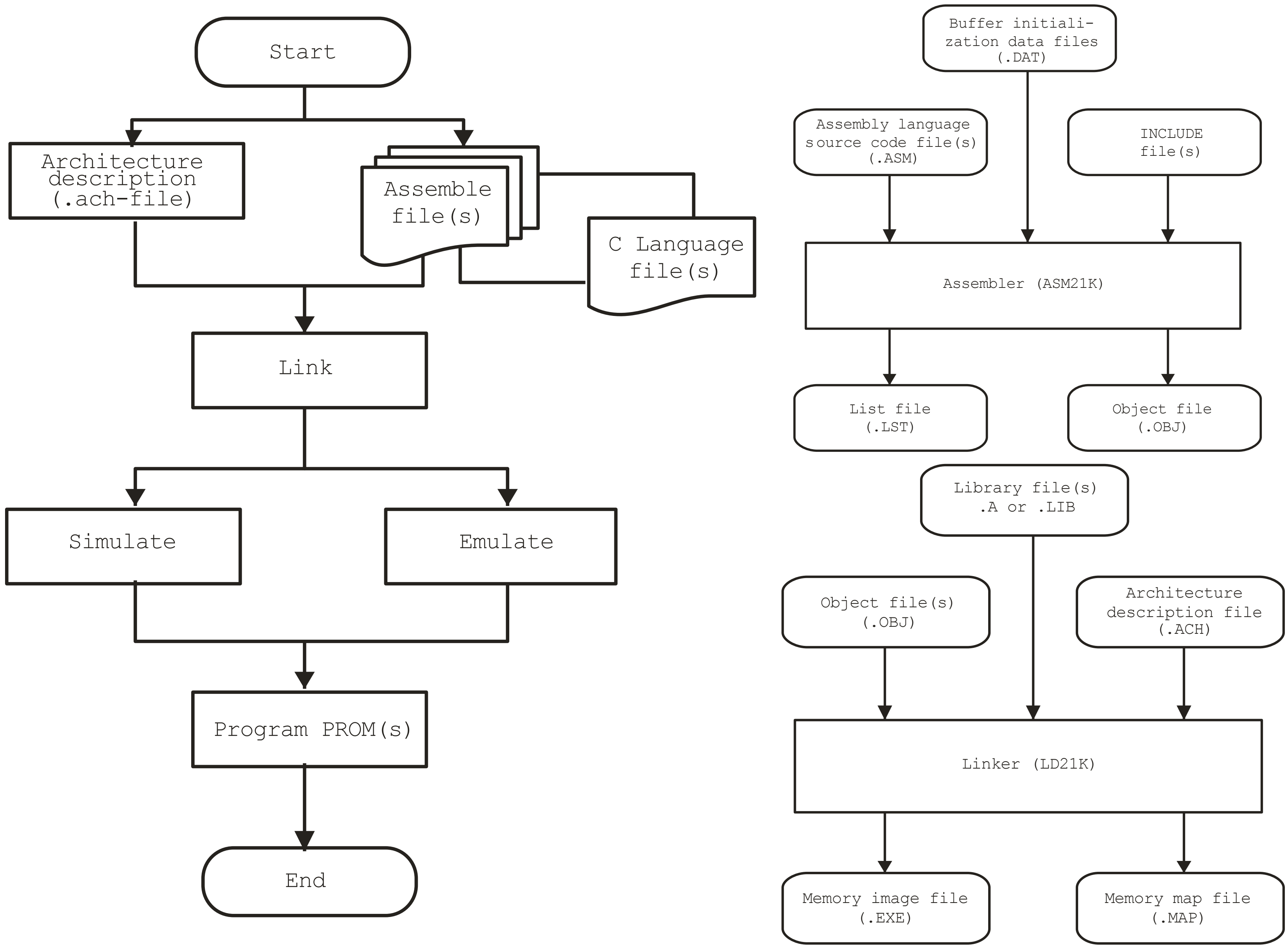


	Instruction	SZ	SV	SS		Instruction	SZ	SV	SS
		Flags							
c	Rn = LSHIFT Rx BY Ry	*	*	0		Rn = FDEP Rx BY Ry	*	*	0
c	Rn = LSHIFT Rx BY <data8>	*	*	0		Rn = FDEP Rx BY <bit6>:<len6>	*	*	0
c	Rn = Rn OR LSHIFT Rx BY Ry	*	*	0		Rn = Rn OR FDEP Rx BY Ry	*	*	0
c	Rn = Rn OR LSHIFT Rx BY <data8>	*	*	0		Rn = Rn OR FDEP Rx BY <bit6>:<len6>	*	*	0
c	Rn = ASHIFT Rx BY Ry	*	*	0		Rn = FDEP Rx BY Ry (SE)	*	*	0
c	Rn = ASHIFT Rx BY<data8>	*	*	0		Rn = FDEP Rx BY <bit6>:<len6> (SE)	*	*	0
c	Rn = Rn OR ASHIFT Rx BY Ry	*	*	0		Rn = Rn OR FDEP Rx BY Ry (SE)	*	*	0
c	Rn = Rn OR ASHIFT Rx BY <data8>	*	*	0		Rn = Rn OR FDEP Rx BY <bit6>:<len6> (SE)	*	*	0
	Rn = ROT Rx BY RY	*	0	0		Rn = FEXT Rx BY Ry	*	*	0
	Rn = ROT Rx BY <data8>	*	0	0		Rn = FEXT Rx BY <bit6>:<len6>	*	*	0
	Rn = BCLR Rx BY Ry	*	*	0		Rn = FEXT Rx BY Ry (SE)	*	*	0
	Rn = BCLR Rx BY <data8>	*	*	0		Rn = FEXT Rx BY <bit6>:<len6> (SE)	*	*	0
	Rn = BSET Rx BY Ry	*	*	0	c	Rn = EXP Rx (EX)	*	0	*
	Rn = BSET Rx BY <data8>	*	*	0	c	Rn = EXP Rx	*	0	*
	Rn = BTGL Rx BY Ry	*	*	0		Rn = LEFTZ Rx	*	*	0
	Rn = BTGL Rx BY <data8>	*	*	0		Rn = LEFTO Rx	*	*	0
	BTST Rx BY Ry	*	*	0		Rn = FPACK Fx	0	*	0
	BTST Rx BY <data8>	*	*	0		Fn = FUNPACK Rx	0	0	0

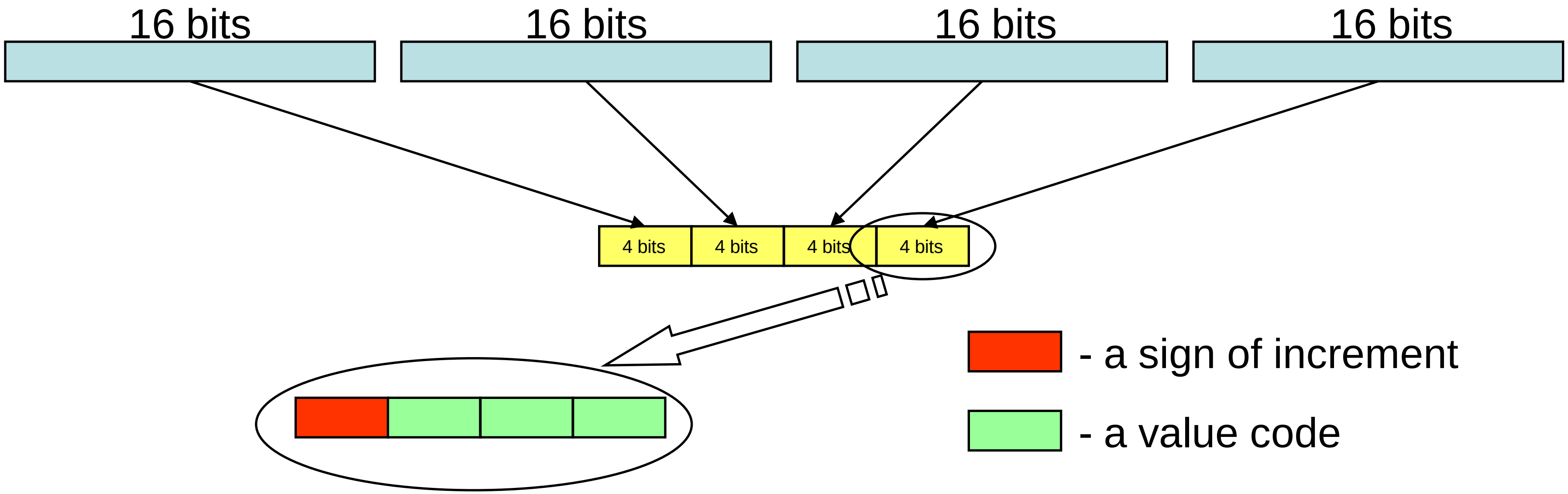
Instruction		*ASTAT Flags*				*STKY Flags*			
		MU	MN	MV	MI	MUS	MOS	MVS	MIS
Fixed-Point:									
Rn MRF MRB = Rx * Ry	(S/U S/U F/I/FR)	*	*	*	0	–	**	–	–
Rn = MRF Rn = MRB MRF = MRF MRB = MRB + Rx * Ry	(S/U S/U F/I/FR)	*	*	*	0	–	**	–	–
Rn = MRF Rn = MRB MRF = MRF MRB = MRB – Rx * Ry	(S/U S/U F/I/FR)	*	*	*	0	–	**	–	–
Rn = SAT MRF Rn = SAT MRB MRF = SAT MRF MRB = SAT MRB	(SI) (UI) (SF) (UF)	*	*	*	0	–	**	–	–
Rn = RND MRF Rn = RND MRB MRF = RND MRF MRB = RND MRB	(SF) (UF)	*	*	*	0	–	**	–	–
MRF MRB = 0		0	0	0	0	–	–	–	–
MRxF MRxB = Rn		0	0	0	0	–	–	–	–
Rn = MRxF MRxB		0	0	0	0	–	–	–	–
Floating-Point:									
Fn = Fx * Fy		*	*	*	*	**	–	**	**

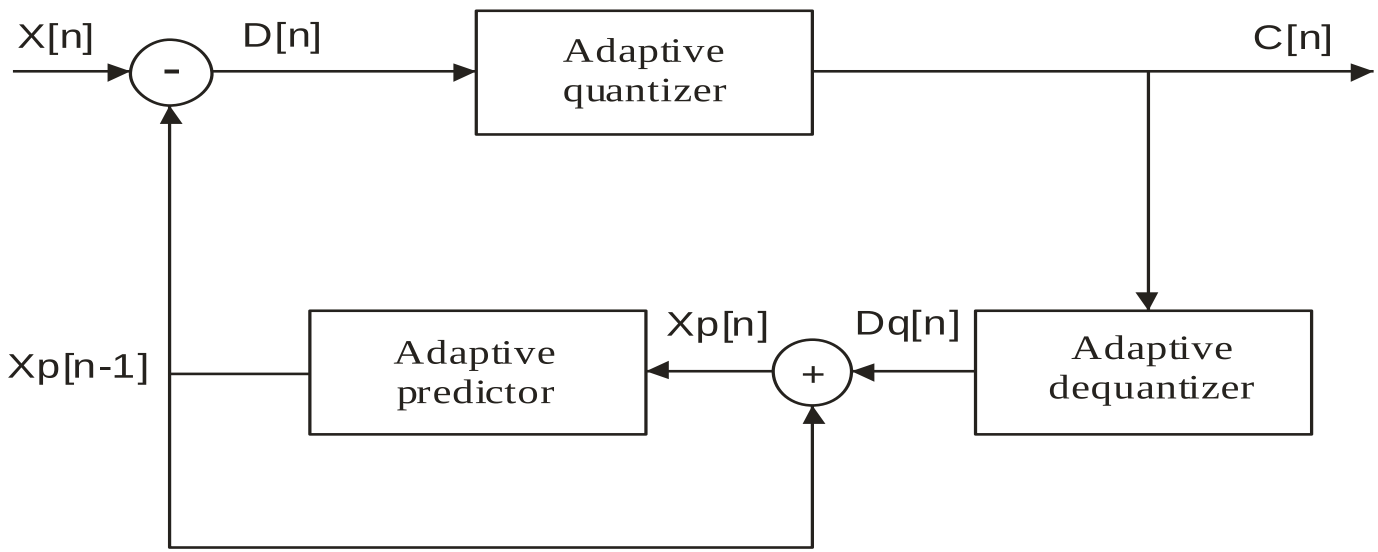
	Instruction	AZ	AV	AN	AC	AS	AI	AF	CACC	AUS	AVS	AOS	AIS
		ASTAT Status Flags								*STKY Status Flags*			
	Fixed-point:												
c	Rn = Rx + Ry	*	*	*	*	0	0	0	–	–	–	**	–
c	Rn = Rx – Ry	*	*	*	*	0	0	0	–	–	–	**	–
c	Rn = Rx + Ry + CI	*	*	*	*	0	0	0	–	–	–	**	–
c	Rn = Rx – Ry + CI – 1	*	*	*	*	0	0	0	–	–	–	**	–
	Rn = (Rx + Ry)/2	*	0	*	*	0	0	0	–	–	–	–	–
	COMP(Rx, Ry)	*	0	*	0	0	0	0	*	–	–	–	–
	Rn = Rx + CI	*	*	*	*	0	0	0	–	–	–	**	–
	Rn = Rx + CI – 1	*	*	*	*	0	0	0	–	–	–	**	–
	Rn = Rx + 1	*	*	*	*	0	0	0	–	–	–	**	–
	Rn = Rx – 1	*	*	*	*	0	0	0	–	–	–	**	–
c	Rn = –Rx	*	*	*	*	0	0	0	–	–	–	**	–
c	Rn = ABS Rx	*	*	0	0	*	0	0	–	–	–	**	–
	Rn = PASS Rx	*	0	*	0	0	0	0	–	–	–	–	–
c	Rn = Rx AND Ry	*	0	*	0	0	0	0	–	–	–	–	–
c	Rn = Rx OR Ry	*	0	*	0	0	0	0	–	–	–	–	–
c	Rn = Rx XOR Ry	*	0	*	0	0	0	0	–	–	–	–	–
c	Rn = NOT Rx	*	0	*	0	0	0	0	–	–	–	–	–
	Rn = MIN(Rx, Ry)	*	0	*	0	0	0	0	–	–	–	–	–
	Rn = MAX(Rx, Ry)	*	0	*	0	0	0	0	–	–	–	–	–
	Rn = CLIP Rx BY Ry	*	0	*	0	0	0	0	–	–	–	–	–
	Floating-point:												
	Fn = Fx + Fy	*	*	*	0	0	*	1	–	**	**	–	**
	Fn = Fx – Fy	*	*	*	0	0	*	1	–	**	**	–	**
	Fn = ABS (Fx + Fy)	*	*	0	0	0	*	1	–	**	**	–	**
	Fn = ABS (Fx – Fy)	*	*	0	0	0	*	1	–	**	**	–	**
	Fn = (Fx + Fy)/2	*	0	*	0	0	*	1	–	**	–	–	**
	COMP(Fx, Fy)	*	0	*	0	0	*	1	*	–	–	–	**
	Fn = –Fx	*	*	*	0	0	*	1	–	–	**	–	**
	Fn = ABS Fx	*	*	0	0	*	*	1	–	–	**	–	**
	Fn = PASS Fx	*	0	*	0	0	*	1	–	–	–	–	**
	Fn = RND Fx	*	*	*	0	0	*	1	–	–	**	–	**
	Fn = SCALB Fx BY Ry	*	*	*	0	0	*	1	–	**	**	–	**
	Rn = MANT Fx	*	*	0	0	*	*	1	–	–	**	–	**
	Rn = LOGB Fx	*	*	*	0	0	*	1	–	–	**	–	**
	Rn = FIX Fx BY Ry	*	*	*	0	0	*	1	–	**	**	–	**
	Rn = FIX Fx	*	*	*	0	0	*	1	–	**	**	–	**
	Fn = FLOAT Rx BY Ry	*	*	*	0	0	0	1	–	**	**	–	–
	Fn = FLOAT Rx	*	0	*	0	0	0	1	–	–	–	–	–
	Fn = RECIPS Fx	*	*	*	0	0	*	1	–	**	**	–	**
	Fn = RSQRTS Fx	*	*	*	0	0	*	1	–	–	**	–	**
	Fn = Fx COPYSIGN Fy	*	0	*	0	0	*	1	–	–	–	–	**
	Fn = MIN(Fx, Fy)	*	0	*	0	0	*	1	–	–	–	–	**
	Fn = MAX(Fx, Fy)	*	0	*	0	0	*	1	–	–	–	–	**
	Fn = CLIP Fx BY Fy	*	0	*	0	0	*	1	–	–	–	–	**

Creation of the digital signal processing applications

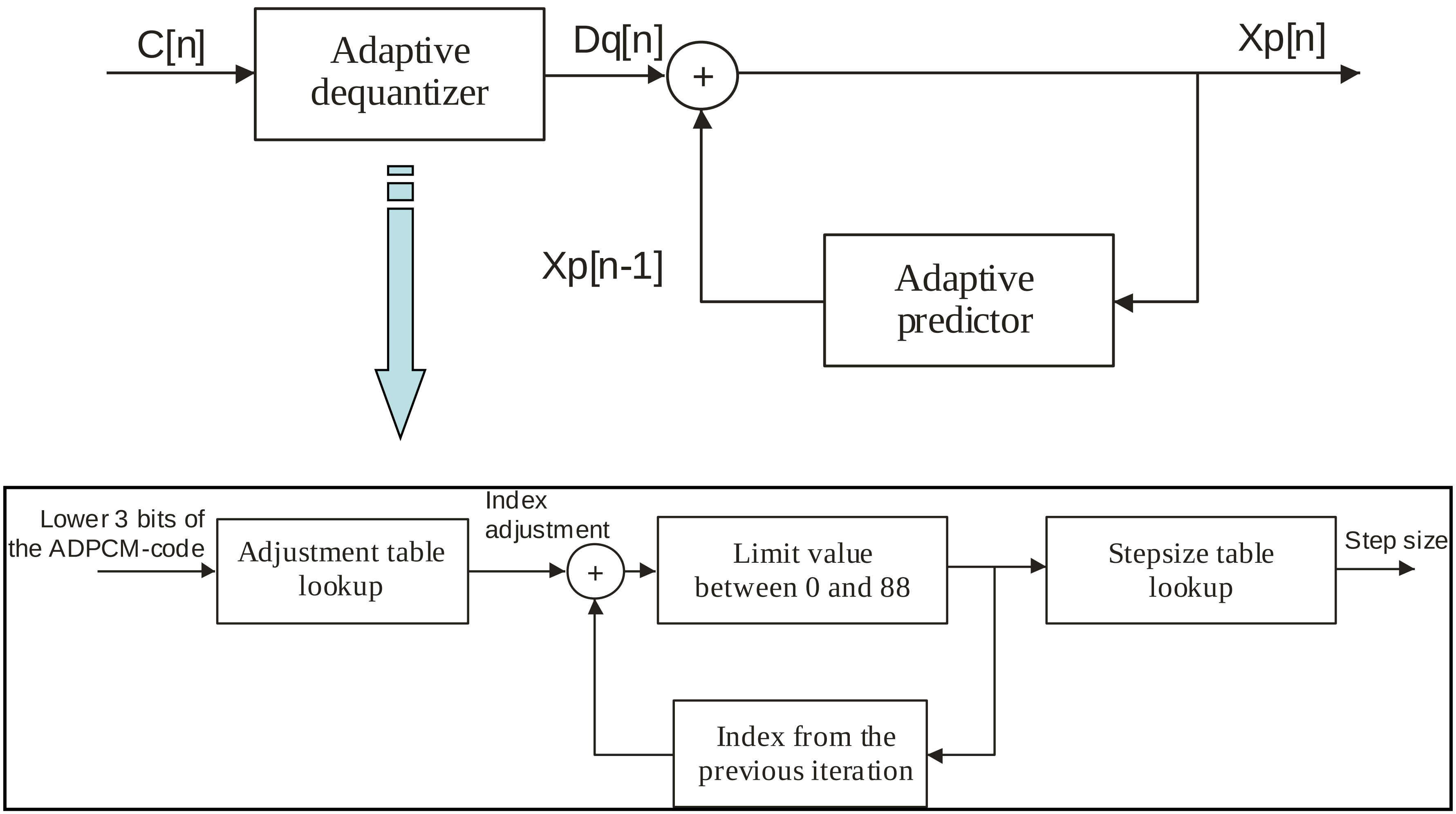


The IMA ADPCM audiodata compression algorithm





Data decoding in the IMA ADPCM algorithm



Mathematical description of the IMA ADPCM

Coding:

$$X_0 = 0$$

$$I_1 = 0$$

$$D_n = X_n - X_{n-1}$$

$$C_n = \frac{4 \cdot D_n}{Si[I_n]}$$

$$I_{n+1} = I_n + Ia[|C_n|]$$

Decoding:

$$X_0 = 0$$

$$I_1 = 0$$

$$D_n = \frac{(C_n + 0.5) \cdot Si[I_n]}{4}$$

$$X_n = X_{n-1} + D_n$$

$$I_{n+1} = I_n + Ia[|C_n|]$$

Si:

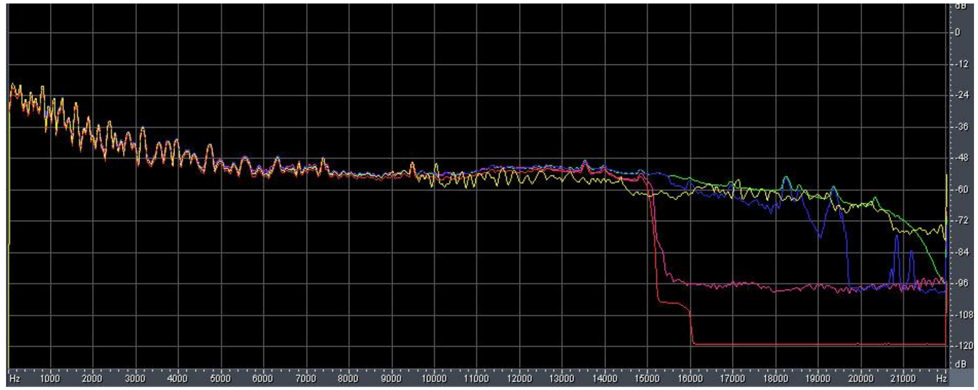
7,	8,	9,	10,	11,	12,	13,	14,	16,
17,	19,	21,	23,	25,	28,	31,	34,	37,
41,	45,	50,	55,	60,	66,	73,	80,	88,
97,	107,	118,	130,	143,	157,	173,	190,	209,
230,	253,	279,	307,	337,	371,	408,	449,	494,
544,	598,	658,	724,	796,	876,	963,	1060,	1166,
1282,	1411,	1552,	1707,	1878,	2066,	2272,	2499,	2749,
3024,	3327,	3660,	4026,	4428,	4871,	5358,	5894,	6484,
7132,	7845,	8630,	9493,	10442,	11487,	12635,	13899,	15289,
16818,	18500,	20350,	22385,	24623,	27086,	29794,	32767	

Ia:

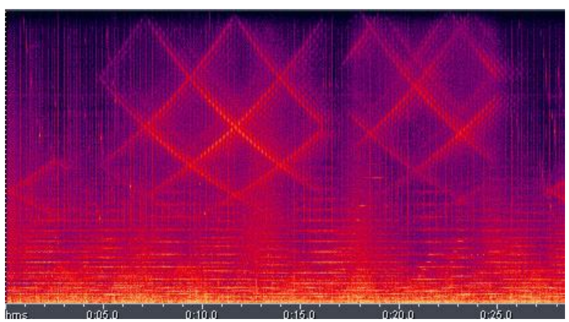
```
-1, -1, -1, -1,
 2,  4,  6,  8
```

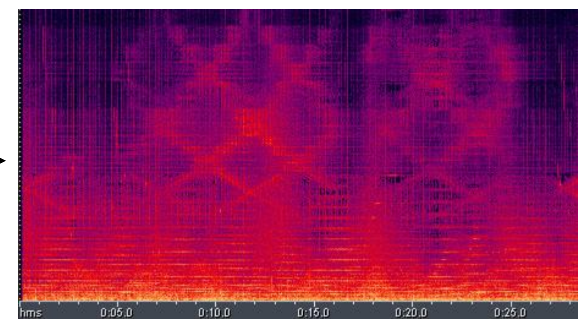
Methods for quality testing of various compression algorithms

The average AFR comparison



The voiceprints comparison





«Blind testing»

The method implies subjective comparison of a sounding quality. The tested signal is represented in two variants: the original and the compressed signal. The user attentively listens the original signal once. Then the user "blindly" listens the unknown order of N original signals and N compressed signals. During each separate listening the user makes a decision, if he listens to the original signal or not, writing down the estimation. Upon the end of the testing the correct/incorrect decisions ratio is evaluated. The closer this ratio is to 1, the less different the original and compressed signals are.

Conclusions and results

- Performed the analysis of methods and formats of a digital sound representation
- Existed methods of digital audiostreams compression investigated
- Implemented the laboratory complex and developed methodology for creating applications of digital audiostreams compression and use of the complex
- Implemented the algorithm for audiostream compression and decompression, providing 4-time volume reduction for the audiodata in real time
- Methods for quality testing of various algorithms for audiostreams compression are offered

- On the theme of this work 3 articles were announced in science conference reports corpus and in periodical magazine, also the study guide was published