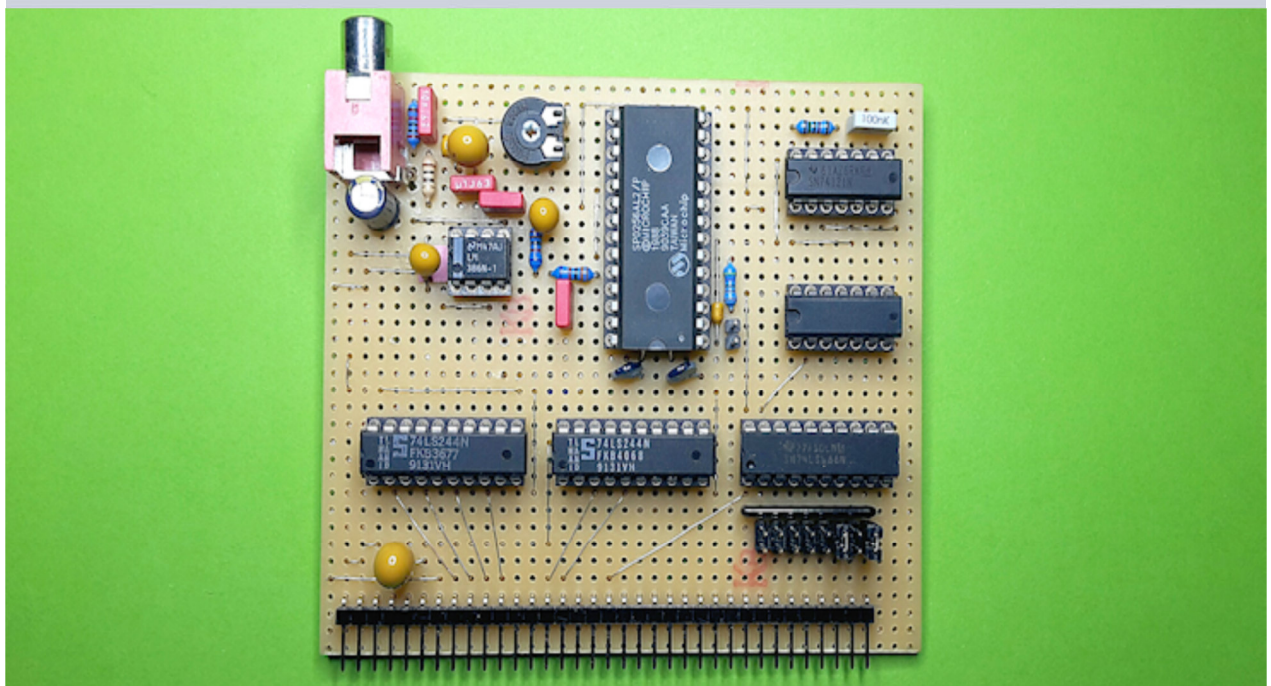


Spezifikation

Speech 2 – eine neue Sprachbaugruppe mit 68008-Programm für den NDR-Klein-Computer



Version 1.0

Idee:

Sascha Neuschl
 Pirolweg 21
 48167 Münster
 Email: scn69@gmx.de

Dokumentenhistorie

Version	Autor(en)	Änderung	Datum
1.0	Neuschl, Sascha	Erste Version	09.02.2020

Inhaltsverzeichnis

1	Vorwort.....	4
1.1	Idee.....	4
1.2	Ansatz.....	4
1.3	Aktueller Stand	4
2	Beschreibung des Konzepts	4
3	Schaltungsprinzip.....	5
3.1	Adresslogik	5
3.2	Datenbuspuffer	6
3.3	Speech-Prozessor SP0256-AL2.....	6
4	Schaltplan, Layout, Bestückungsplan und Stückliste NKC-Speech 2-Karte	7
4.1	Schaltplan:	7
4.2	Layout:	8
4.3	Bestückungsplan (vorläufig):	8
4.4	Stückliste:	9
5	Anmerkungen.....	10
5.1	Eingänge und Ausgänge:	10
5.2	Allgemeines Verhalten:.....	10
6	Aufbau und Test der Speech 2 - Baugruppe	11
6.1	Testprogramm	11
7	Anhang.....	16
7.1	Datenblätter TTL-Bausteine:	16
7.1.1	74LS688	16
7.1.2	74LS244	16
7.1.3	74S32	17
7.1.1	74121	17
7.2	SP0256-AL2:	18
7.3	Verweis auf Datenblätter komplexer Bausteine und Spezifikationen / Quellennachweis	19

1 Vorwort

1.1 Idee

Bei ebay hatte ich nach Sprachsynthesizer-ICs gesucht und der Chip SP0256 – AL2 von ursprünglich General Instruments war mir über den Weg gelaufen. Er war günstig aus China zu haben und schon war das Projekt „**Speech 2**“ geboren.

Allerdings gab es dann doch noch einen Stolperstein:

Man achte darauf, dass man wirklich einen SP0256 – AL2 bekommt und nicht die Version „012“ - an der Rückseite des Chips angedruckt. Die Version „012“ ist eine besondere Produktvariante, die wenige, ganze Wörter im internen ROM gespeichert hat und nicht Allophones, aus denen man Wörter bilden kann.

1.2 Ansatz

Hier war nichts Besonderes zu beachten, da ja nur ein I/O-Baustein an den NKC anzuschließen war.

1.3 Aktueller Stand

Es liegt keine industrielle Platine für den Nachbau vor.

Das Testprogramm liegt in Version 1.0 vor.

2 Beschreibung des Konzepts

Das Konzept der **Speech 2 – Baugruppe** besteht darin, den **Speech-Prozessor SP0256-AL2** mit einer einzigen I/O-Adresse an den NKC-Bus anzuschließen und dabei einerseits Daten schreiben und den Status lesen zu können.

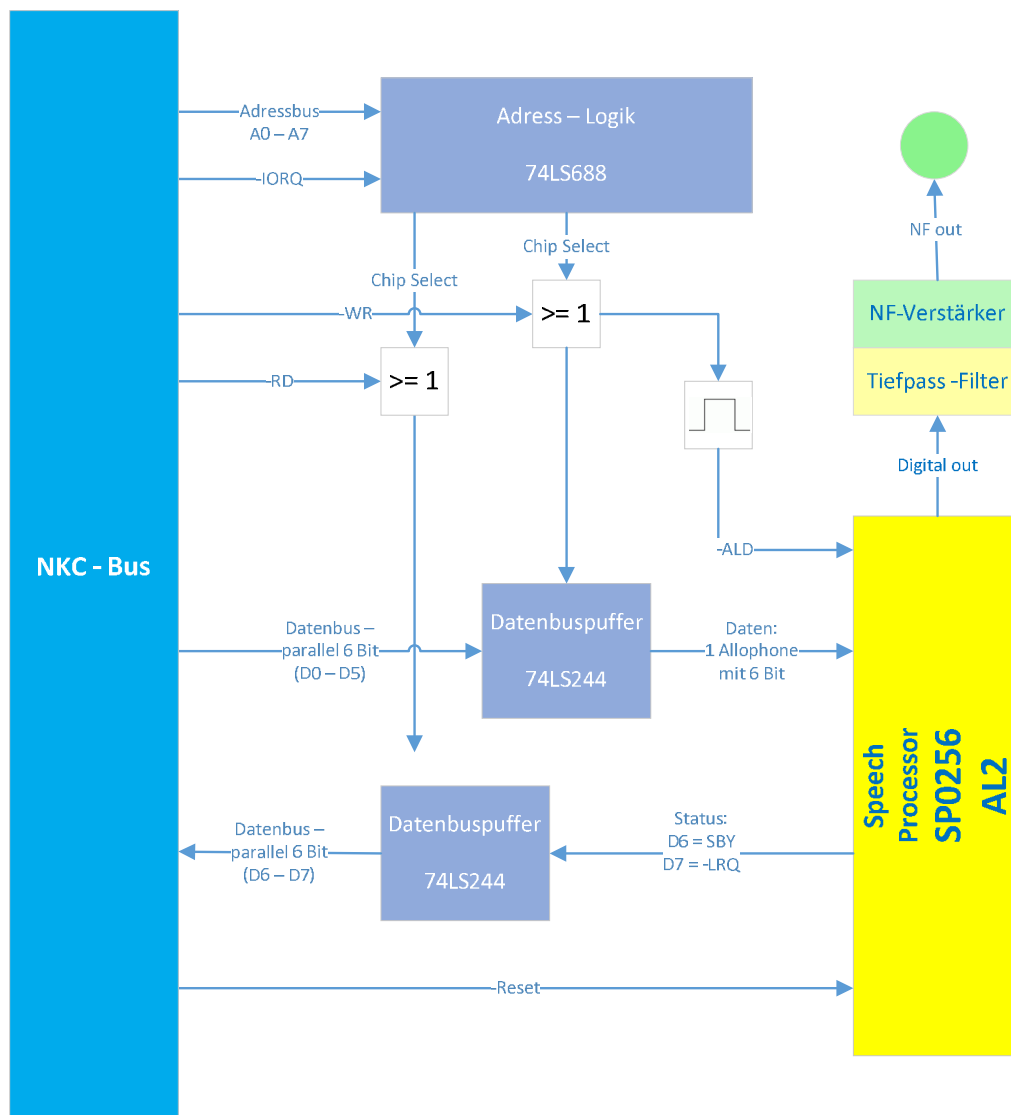
Der **digitale Ausgang** wird über ein **Tiefpass-Filter** an einen **NF-Verstärker** geleitet, an den entweder direkt ein Lautsprecher angeschlossen oder ein nachfolgender Verstärker angeschlossen werden können.

Das **Prinzip der Spracherzeugung** des **SP0256-AL2** ist, dass **64 Laute - genannt Allophones -**, die im internen ROM des Bausteins gespeichert sind, auszugeben, sodass **Wörter** entstehen.

Folgende aktive Komponenten werden eingesetzt:

Funktion	Baustein	Beschreibung	Datenblatt
Speech-Prozessor	Sp0256-AL2	Es handelt sich hier um einen Sprach-Synthesizer, bei dem man mit sogenannten Allophones Wörter bilden und ausgeben kann.	SP0256_AL2.pdf
NF-Verstärker	LM386	NF-Verstärker	LM386.pdf

3 Schaltungsprinzip



3.1 Adresslogik

Die Adresslogik wird mit dem Baustein **74LS688** abgebildet. Er benötigt zur Freigabe das **IO-Request-Signal (-IORQ)** des NKC-Busses. Des Weiteren werden **die Adressleitungen A0 bis A7** zugeführt und mit den Werten, die am **Adressjumper** (JMP 1 - siehe Schaltplan) eingestellt werden, verglichen. Am **Adressjumper** wird somit die **IO-Adresse** des Speech-Prozessors **SP0256-AL2** eingestellt. Da wir nur **eine einzige I/O-Adresse** benötigen, werden alle Adressleitungen A0-A7 an den **74LS688** geführt. Sind die angelegten Adressdaten des **Adressjumpers** und der **Adressleitungen A2 bis A7** identisch, dann wird das Signal **-Chip Select** ausgelöst. Dieses Signal wird mit jeweils einem Oder-Gatter des Bausteins **74LS32** mit den Steuerleitungen **-RD** und **-WR** kombiniert, so dass jeweils ein **-Chip Select**-Signal für das **Lesen** vom und das **Schreiben** auf den **SP0256-AL2** entsteht. Das **-Chip Select**-Signal für das **Schreiben** gelangt außerdem auf den Baustein **74121**. Dieses Monoflop verlängert die Impulsdauer des Signals. Es wird dann an den PIN **-ALD** (address load) des **SP0256-AL2** geführt.

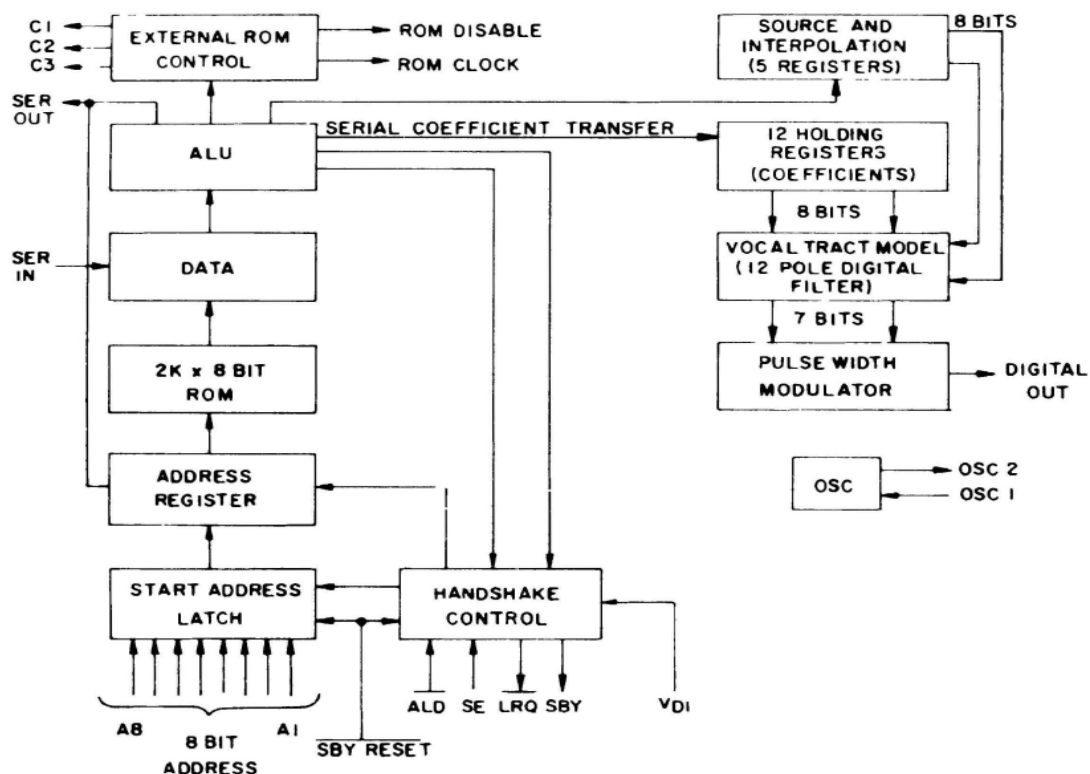
3.2 Datenbuspuffer

Als **Datenpuffer** werden zwei **74LS244** eingesetzt. Die Freigabe erfolgt durch die Signale **–Chip Select** für **Lesen** und **Schreiben**. Über den **Datenbuspuffer zum Schreiben** werden die **Datenbits D0 bis D5** vom **NKC-Bus** übertragen. Obwohl der **SP0256-AL2** eigentlich 8 Datenleitungen besitzt, genügt es nur 6 davon zu benutzen, weil das interne ROM nur 64 Allophones beinhaltet. Die Datenleitungen A7 und A8 werden auf 0 Volt gelegt. Über den **Datenbuspuffer zum Lesen** werden die **Datenbits D6 bis D7** zum **NKC-Bus** übertragen.

3.3 Speech-Prozessor SP0256-AL2

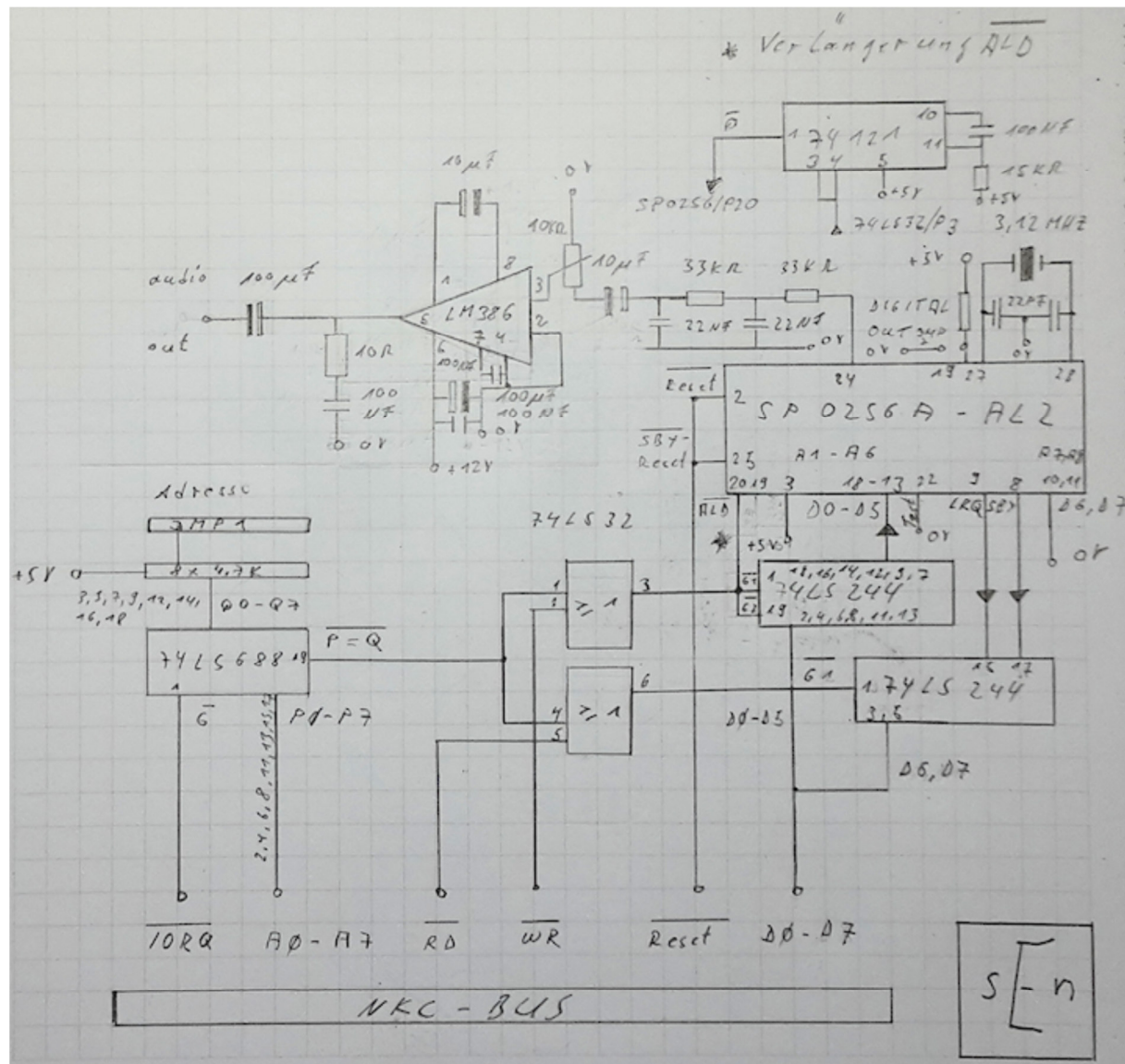
Bei einem Schreibvorgang auf den Speech-Prozessor **SP0256-AL2** wird durch das Signal **–Chip Select zum Schreiben** zunächst der entsprechende **Datenbuspuffer freigegeben** und die **Daten** werden angestellt. **Danach** wird durch das Signal **–Chip Select zum Schreiben mit Impulsverlängerung** auf dem PIN **–ALD** (address load) der Speech-Prozessor freigegeben. Das gilt dann, wenn der PIN **SE** (strobe enable) auf „logisch 1“ liegt. **Dieser Modus ist in der Schaltung und im Testprogramm realisiert!** Liegt dieser PIN auf „logisch 0“, dann übernimmt der **SP0256-AL2** Daten, wenn **eine oder mehrere seiner Datenleitungen** (im Datenblatt Adressleitungen genannt) einen **Übergang von „logisch 0“ zu „logisch 1“** aufweist. Es gibt einen Jumper **JMP 2**, mit dem der Zustand von **SE** eingestellt werden kann. Hat der Speech-Prozessor die **Daten D0 bis D5** übernommen, sucht er im **internen ROM** das zugehörige **Allophone** und gibt es aus. Während dieser Zeit können keine weiteren Daten geschrieben werden. Um zu prüfen, ob der Speech-Prozessor wieder Daten empfangen kann, gibt es **2 Status-Signale**. **–LRQ** (load request) gibt an, dass weitere Daten geschrieben werden können. **SBY** (standby) gibt an, dass der Speech-Prozessor „nichts tut“ bzw. das letzte Allophone ausgegeben hat. Es reicht, das Signal **SBY** abzufragen, weil das Datenschieben sehr schnell, aber die Ausgabe eines **Allophones** langsam geht. In der Schaltung können beide Status-Signale abgefragt werden. **–RESET** des NK-Busses ist an die PINs **–RESET** und **–SBY-RESET** geführt.

SP0256 BLOCK DIAGRAM



4 Schaltplan, Layout, Bestückungsplan und Stückliste NKC-Speech 2-Karte

4.1 Schaltplan:



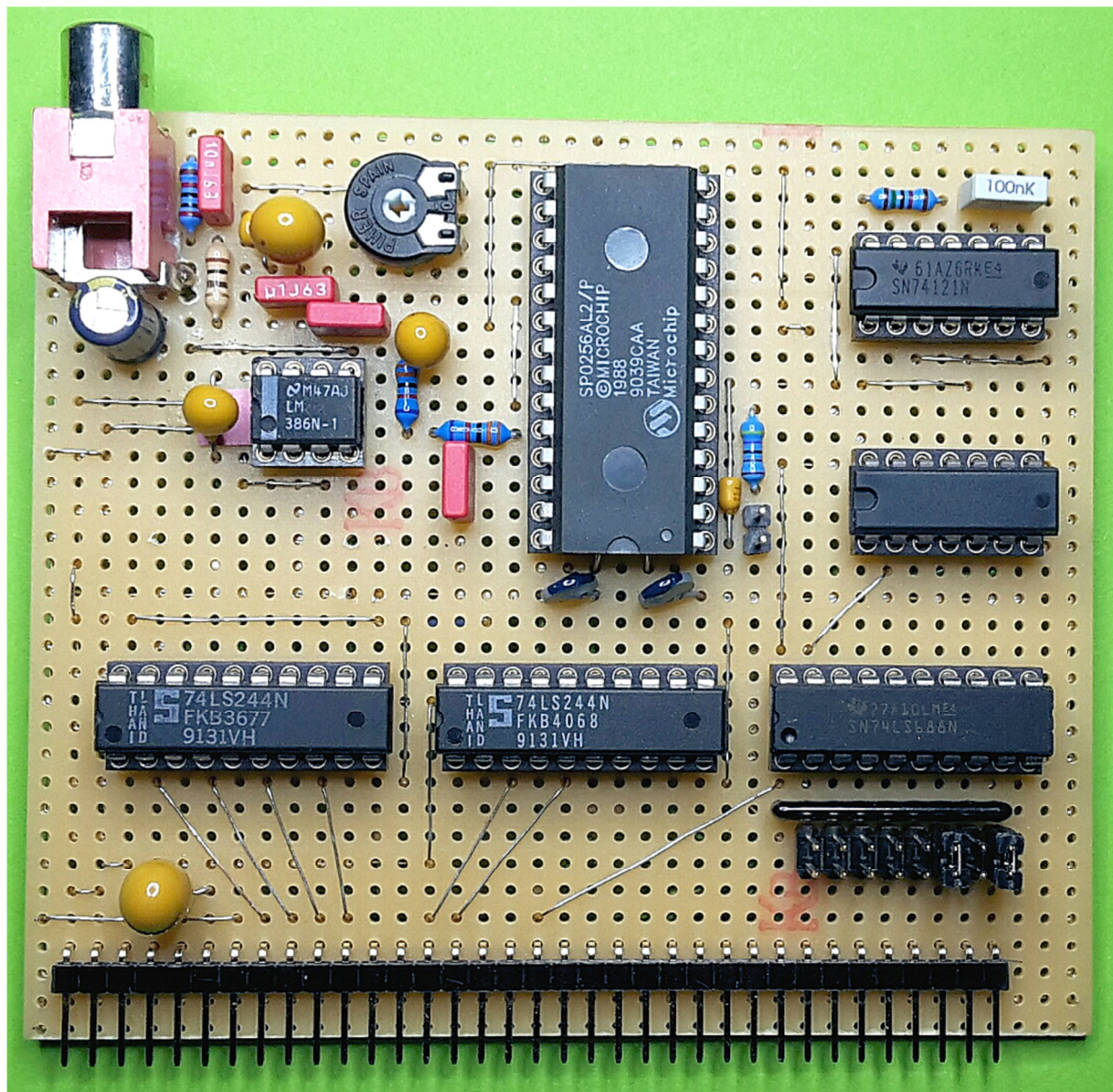
Speech 2 – eine neue Sprachbaugruppe für den NDR-Klein-Computer

4.2 Layout:

Bestückungsseite: Nicht verfügbar

Rückseite: Nicht verfügbar

4.3 Bestückungsplan (vorläufig):



4.4 Stückliste:

Mechanisch:

Lochrasterplatine 10 cm X 9 cm	1 Stück
Stiftleiste - einreihig, abgewinkelt, 35 polig	1 Stück
Stiftleiste - zweireihig, gerade, 8 polig	1 Stück
Stiftleiste - einreihig, gerade, 2 polig	1 Stück
IC-Fassung - 14 polig	2 Stück
IC-Fassung - 20 polig	3 Stück
IC-Fassung - 28 polig	1 Stück
Chinch-Buchse	1 Stück

Widerstände (1/4 Watt) / Trimmer:

10 Ω	1 Stück
2,2 K Ω	1 Stück
4,7 K Ω	1 Stück
15 K Ω	1 Stück
33 K Ω	2 Stück
4,7 K X 8 - Netzwerk	1 Stück
10 K Ω - Trimmer	1 Stück

Kondensatoren:

22 pF - Keramik	2 Stück
10 nF - Folie	1 Stück
22 nF - Folie	2 Stück
100 nF - Keramik	7 Stück
100 nF - Folie)	2 Stück
1 μ F	2 Stück
10 μ F	2 Stück
100 μ F	1 Stück

TTL:

74 LS 32	1 Stück
74 121	1 Stück
74 LS 244	2 Stück
74 LS 688	1 Stück

Analog:

LM386	1 Stück
-------	---------

Speech Prozessor:

SP0256 – AL2	1 Stück
--------------	---------

5 Anmerkungen

5.1 Eingänge und Ausgänge:

Hier werden alle Ein- und Ausgänge sowie Steckverbinder und Jumper der NKC-Speech 2-Karte und beschrieben:

Steckverbinder	Beschreibung	Bemerkung
Adressjumper (JMP1)	Jumper zur Einstellung der NKC-IO-Adresse des Speech-Prozessors SP0256-AL2	Im Testprogramm ist die IO-Adresse \$FFFFFF5F verwendet.
Strobe Enable (JMP2)	Offen = enabled, geschlossen = disabled	Steuert die Datenübernahme Mittels Strobe-Signal oder nur durch Änderung der Zustände der Datenleitungen
Chinch-Buchse	Audio-Ausgang	In der Schaltung mit 2,2 K Ω abgeschlossen für „line in“

5.2 Allgemeines Verhalten:

Beim Betrieb der NKC-Speech 2-Karte traten keine instabilen Situationen auf.

6 Aufbau und Test der Speech 2 - Baugruppe

Zurzeit existiert noch keine industrielle Platine, sodass der Aufbau auf einer Lochrasterplatine erfolgen muss. Um am Ende etwas hören zu können, muss die Chinch-Buchse mit dem Eingang eines Verstärkers verbunden werden.

6.1 Testprogramm

Das Testprogramm arbeitet folgendermaßen:

1. Die hier eingetragene Basisadresse „basis“ ist \$FFFFFF5F und kann natürlich geändert werden.
2. Die Tabelle „taballo“ enthält alle Allophones mit ihrem Symbol und ihrem Wert (Adresse im internen ROM des SP0256-AL2)
3. Möchte man ein Wort oder einen Satz mit mehreren Worten ausgeben, so kombiniert man die Symbole der passenden Allophones in einem String. Dies wurde hier in der Variablen „word“ für das Wort „Computer“ getan.
4. Die Variable „word“ wird in den Symbol-Puffer „buffsym“ geladen. Mit dem Unterprogramm „convert“ wird einerseits der String im Symbolpuffer geparkt und parallel der Werte-Puffer „buffval“ geschrieben.
5. Der Inhalt des Werte-Puffers wird mit dem Unterprogramm „writesp“ Byte für Byte an den speech-Prozessor übergeben. Dabei wird vor dem Schreiben der Daten das Signal SBY abgefragt. Ist es auf „logisch 1“ dürfen weitere Daten geschrieben werden.
6. Läuft das Programm fehlerfrei durch, sollte am Audioausgang des Verstärkers das Wort „Computer“ zu hören sein.

```

*****
*                               *
*       Test Speech 2 mit dem Chip SP0256A-AL2                       *
*                               *
*****
* - Basisadresse = $FFFFFF5F                                         *
* - Es wird nur eine Adresse benötigt - zum Lesen und zum Schreiben *
* - Geschrieben werden nur 6 Bit, die beiden höchstwertigen Adress- *
*   Pins werden auf 0 Volt gelegt.                                   *
* - Gelesen werden die Signale SBY und -LRQ, für das Handshake beim *
*   Schreiben der Daten.                                           *
*****
*                               *
*       Version 1.0 - 10.02.2020 - by sEn                           *
*                               *
*****

start:
    bra sprache
    rts

```

Speech 2 – eine neue Sprachbaugruppe für den NDR-Klein-Computer

; * Konstanten

; Adresse:

```

basis    equ    $ffffff5f    ; Adresse Sprachchip
                                ; Daten schreiben = Bit 0-5
                                ; Handshake Bit 6 = sby, Bit 7 = -LRQ

numrow   equ    64           ; Anzahl Zeilen der tabelle taballo
                                ; mit Allophones

```

; * Variablen

taballo: ; Allophone - Tabelle

```

dc.b 'pa1/', $00    ; pause - 10 ms
dc.b 'pa2/', $01    ; pause - 20 ms
dc.b 'pa3/', $02    ; pause - 50 ms
dc.b 'pa4/', $03    ; pause 100 ms
dc.b 'pa5/', $04    ; pause 200 ms
dc.b 'oi1/', $05    ; boy
dc.b 'ay1/', $06    ; sky
dc.b 'eh1/', $07    ; end
dc.b 'kk3/', $08    ; comb
dc.b 'pp1/', $09    ; pow
dc.b 'jh1/', $0a    ; dodge
dc.b 'nn1/', $0b    ; thin
dc.b 'ih1/', $0c    ; sit
dc.b 'tt2/', $0d    ; to
dc.b 'rr1/', $0e    ; rural
dc.b 'ax1/', $0f    ; succeed
dc.b 'mm1/', $10    ; milk
dc.b 'tt1/', $11    ; part
dc.b 'dh1/', $12    ; they
dc.b 'iy1/', $13    ; see
dc.b 'ey1/', $14    ; beige
dc.b 'dd1/', $15    ; could
dc.b 'uw1/', $16    ; to
dc.b 'ao1/', $17    ; aught
dc.b 'aa1/', $18    ; hot
dc.b 'yy2/', $19    ; yes
dc.b 'ae1/', $1a    ; hat
dc.b 'hh1/', $1b    ; he
dc.b 'bb1/', $1c    ; business
dc.b 'th1/', $1d    ; thin

```


Speech 2 – eine neue Sprachbaugruppe für den NDR-Klein-Computer

```

dc.b 'uh1/',$1e      ; book
dc.b 'uw2/',$1f      ; food
dc.b 'aw1/',$20      ; out
dc.b 'dd2/',$21      ; do
dc.b 'gg3/',$22      ; wig
dc.b 'vv1/',$23      ; vest
dc.b 'gg1/',$24      ; got
dc.b 'sh1/',$25      ; ship
dc.b 'zh1/',$26      ; azure
dc.b 'rr2/',$27      ; brain
dc.b 'ff1/',$28      ; food
dc.b 'kk2/',$29      ; sky
dc.b 'kk1/',$2a      ; can't
dc.b 'zz1/',$2b      ; zoo
dc.b 'ng1/',$2c      ; anchor
dc.b 'll1/',$2d      ; lake
dc.b 'ww1/',$2e      ; wool
dc.b 'xr1/',$2f      ; repair
dc.b 'wh1/',$30      ; whig
dc.b 'yy1/',$31      ; yes
dc.b 'ch1/',$32      ; church
dc.b 'er1/',$33      ; fir
dc.b 'er2/',$34      ; fir
dc.b 'ow1/',$35      ; beau
dc.b 'dh2/',$36      ; they
dc.b 'ss1/',$37      ; vest
dc.b 'nn2/',$38      ; no
dc.b 'hh2/',$39      ; hoe
dc.b 'or1/',$3a      ; store
dc.b 'ar1/',$3b      ; alarm
dc.b 'yr1/',$3c      ; clear
dc.b 'gg2/',$3d      ; guest
dc.b 'el1/',$3e      ; saddle
dc.b 'bb2/',$3f      ; business

buffsym:      ds.b 300 ; Symbol-Puffer für Allophones
buffval:      ds.b 300 ; Werte-Puffer für Allophones

;Testwort "computer"
word:         dc.b 'kk1/ax1/mm1/pp1/yy1/uw1/tt2/er1/pa5/****'

ds 0

```

Speech 2 – eine neue Sprachbaugruppe für den NDR-Klein-Computer

;* Unterprogramme

;* Daten schreiben (Übergabe des Wert-Puffers)

writesp:

```

lea buffval, a0      ; Puffer mit allophone Folge als Werte
satzval:             ; Ausgabeschleife
    move.b basis, d0  ; Check, ob SPEECH processor neue Daten annimmt
    and #01000000, d0 ; Nur Bit 6 = SBY entscheidend
    beq.s satzval      ; wenn LRQ = 0, dann ready
    move.b (a0)+, d1   ; allophone-Wert holen
    cmp.b #'@', d1     ; Ende des Werte-Puffers bei '@'
    beq.s fertig
    move.b d1, basis   ; allophone schreiben
    bra.s satzval      ; naechster allophone-Wert
fertig:
rts

```

;* Allophone Symbol in Wert wandeln (Übergabe des Symbol-Puffers)

convert:

```

;lea buffsym, a0      ; Puffer mit Allophone-Symbolen
lea word, a0          ; Testwort "copmputer"
lea buffval, a2       ; Ziel-Puffer mit Allophone-Werten
pars:                 ; Parsen der Symbole im Symbol-Puffer
    ; 4 Bytes für Symbol oder Ende mit '****'

    move.b (a0)+, d1   ; Byte 1
    lsl.l #8, d1
    move.b (a0)+, d1   ; Byte 2
    lsl.l #8, d1
    move.b (a0)+, d1   ; Byte 3
    lsl.l #8, d1
    move.b (a0)+, d1   ; Byte 4
    cmp.l #'****', d1  ; Ende des Symbol-Puffers erkannt?
    beq.s endpars
    lea taballo, a1    ; Symbol erkannt, jetzt suchen in taballo
    move #numrow-1, d3 ; 64 Zeilen
    getsym:
        move.b (a1)+, d2 ; Byte 1
        lsl.l #8, d2
        move.b (a1)+, d2 ; Byte 2

```

Speech 2 – eine neue Sprachbaugruppe für den NDR-Klein-Computer

```

    lsl.l #8, d2
    move.b (a1)+, d2      ; Byte 3
    lsl.l #8, d2
    move.b (a1)+, d2      ; Byte 4
    cmp.l d2, d1          ; wird mit Symbol aus Symbol-Puffer verglichen
    beq.s symfound        ; Wenn gleich, dann Symbol in taballo gefunden
    adda.l #1, a1          ; Adresspointer über Wert zum naechsten Symol
    dbra d3, getsym
    bra.s pars            ; Naechstes Symbol aus Symbol-Puffer
symfound:                ; Symbol aus Puffer in taballo gefunden, jetzt
                        ; Wert dazu ermitteln ...
    move.b (a1)+, (a2)+   ; Wert aus taballo in Werte-Puffer
    bra.s pars            ; Naechstes Symbol aus Symbol-Puffer
endpars:                 ; Das Ende des Symbol-Puffers ist erreicht.
    move.b #'@', (a2)     ; Abschluss für Werte-Puffer
    rts

; * Hauptprogramm

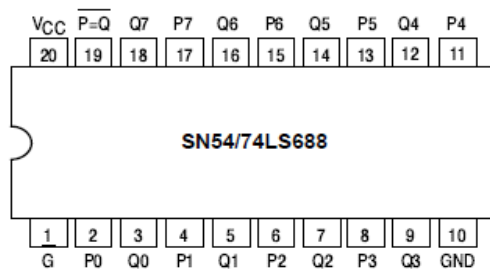
sprache:
    bsr convert
    bsr writesp
    rts

end.
```

7 Anhang

7.1 Datenblätter TTL-Bausteine:

7.1.1 74LS688



TYPE	P = Q	P > Q	OUTPUT ENABLE	OUTPUT CONFIGURATION	PULLUP
LS682	yes	yes	no	totem-pole	yes
LS684	yes	yes	no	totem-pole	no
LS688	yes	no	yes	totem-pole	no

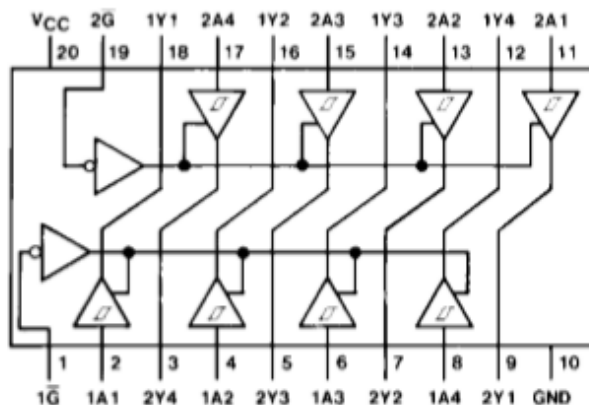
FUNCTION TABLE

INPUTS			OUTPUTS	
DATA	ENABLES		$\overline{P = Q}$	$\overline{P > Q}$
P, Q	$\overline{G}, \overline{GT}$	$\overline{G2}$		
P = Q	L	L	L	H
P > Q	L	L	H	L
P < Q	L	L	H	H
X	H	H	H	H

H = HIGH Level, L = LOW Level, X = Irrelevant

7.1.2 74LS244

Connection Diagram



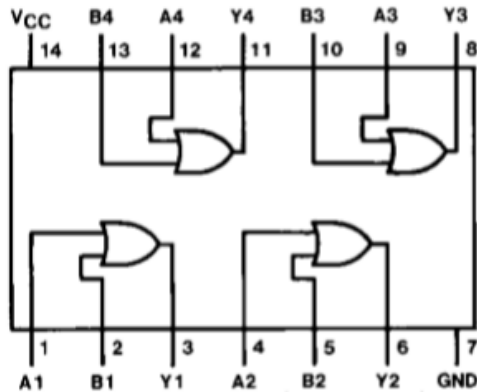
Function Table

Inputs		Output
\overline{G}	A	Y
L	L	L
L	H	H
H	X	Z

L = LOW Logic Level
H = HIGH Logic Level
X = Either LOW or HIGH Logic Level
Z = High Impedance

7.1.3 74S32

Connection Diagram



Function Table

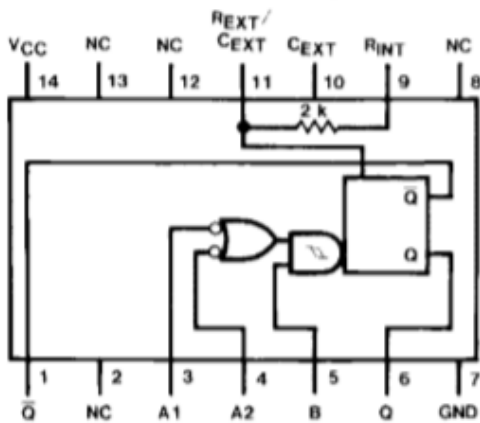
$$Y = A + B$$

Inputs		Output
A	B	Y
L	L	L
L	H	H
H	L	H
H	H	H

H = HIGH Logic Level
L = LOW Logic Level

7.1.1 74121

Connection Diagram



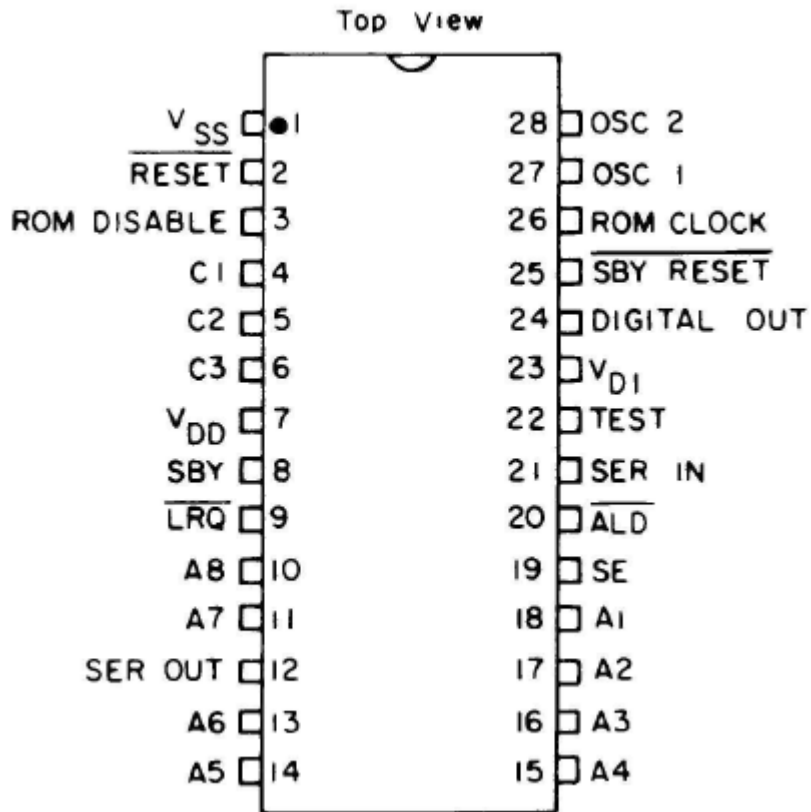
Function Table

Inputs			Outputs	
A1	A2	B	Q	\bar{Q}
L	X	H	L	H
X	L	H	L	H
X	X	L	L	H
H	H	X	L	H
H	↓	H	⌋	⌋
↓	H	H	⌋	⌋
↓	↓	H	⌋	⌋
L	X	↑	⌋	⌋
X	L	↑	⌋	⌋

H = HIGH Logic Level
L = LOW Logic Level
X = Can Be Either LOW or HIGH
⌋ = A Positive Pulse
⌋ = A Negative Pulse

↑ = Positive Going Transition
↓ = Negative Going Transition

7.2 SP0256-AL2:



7.3 Verweis auf Datenblätter komplexer Bausteine und Spezifikationen / Quellennachweis

Baustein/Objekt	Funktion	Datenblatt / Spezifikation
SP0256-AL2	Speech-Prozessor	SP0256_AL2.pdf
LM386	Schnittstelle zu IDE / ATA / ATAPI-Geräten	LM386.pdf
Buch	Buch zum Design mit Speech-Prozessoren	designing_with_speech_processing_chips.pdf