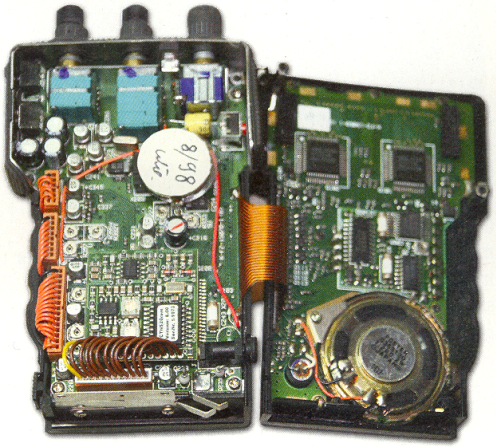
Neues Zubehör für altes Gerät

Remake für das CTN-520

Klaus Hain, DL8NAT

Als vor ca. 20 Jahren die Standard-Funkgeräte C-220, C-228, C-520, C-528, C-620 und C-628 auf den Markt kamen, sparte man sich meist die Anschaffung des optionalen CTCSS-Moduls. Denn CTCSS war in Europa kein Thema. Heute sieht das anders aus.



CTCSS steht für Continuous Tone Coded Squelch System oder Continuous Tone Carrier Subaudio Squelch. Beides kann man frei mit „durch Pilotton gesteuerte Rauschsperre" übersetzen. In den USA wird das System schon lange zum Öffnen der Amateurfunkrelais genutzt und gewinnt in Europa aufgrund zunehmender HF-Störungen immer mehr an Bedeutung.

Ein CTCSS-Decoder/Encoder wird heute in vielen Fällen gewünscht, denn, bedingt durch die in den vergangenen Jahren zunehmenden Störungen, stellen viele Relaisbetreiber (z.B. in der Schweiz) ihre Repeater auf CTCSS-Betrieb um. Daher entstand bei mir der Wunsch, die noch sehr verbreiteten Funkgeräte der Standard-Serie auf CTCSS umzurüsten. Das Aufmacherfoto zeigt den Nachbau in einem C-520. Die Platine ist vollständig mit dem Original CTN-520 kompatibel. Funkamateure, die sich mit dem Thema CTCSS beschäftigen wollen, empfehle ich die Infos auf [1].

Eine eigene Lösung

Von Matthias Tafelmeyer, DG1NMT, der sich u.a. in seiner eigenen Firma mit der Wartung, Pflege und Reparatur der Standard-Funkgeräte beschäftigt, erfuhr ich, dass viele Besitzer diese Geräte mit der CTCSS-Funktion erweitern wollten. Die Originalplatine ist nur noch sehr schwer zu bekommen. Ein direkter Nachbau scheiterte daran, dass das Original-IC S7119A im freien Handel nicht zur Verfügung steht.

Die Suche nach einer fertigen Lösung führte mich zu den Chips der Serie MX335 [2] und FX335 [3]. Das sind komplette CTCSS-De- und Encoder. Die Verfügbarkeit ist aber auch sehr schlecht, hinzu kommen Stückpreise bis 100 €. Als dritter Nachteil unterscheidet sich die Ansteuerung wesentlich von der des S7119A, sodass eine Umcodierung der Ansteuerung notwendig geworden wäre.

Ich entschied mich deshalb zu einer eigenen Lösung. Mein erstes Konzept war: Tiefpass > Schmitt-Trigger > Periodendauermessung > Schaltschwelle > Auswertung. Das Ganze mit einem Mikrocontroller zu bewerkstelligen, sah ich nicht als Problem an.

Doch das erwies sich als großer Irrtum. Einige Versuche auf einer Lochrasterplatte brachten mich sofort wieder auf den Boden der Tatsachen zurück. Das Tiefpassfilter [4] funktionierte zwar auf Anhieb, und der Komparator brachte im Test ein sauberes Signal, doch mit einem CTCSS-Signal (erzeugt mit einem Funktionsgenerator) und einem Sprachsignal – über ein paar Widerstände im richtigen Verhältnis addiert – war das periodische Rechtecksignal verschwunden. Die Flanken entsprachen einer Mischung aus Pilotton und NF. Eine einfache Periodendauermessung war nicht mehr möglich.

Versuche mit PLL und anderen schmalbandigen NF-Filtern zeigten, dass die erforderliche Genauigkeit **(Bild 1)** nur digital erreicht werden kann. Und weil auch der durch die Originalplatine vorgegebene Platz mit 30 mm × 30 mm begrenzt war, entschied ich mich für den Controller PIC16F819.

Zur Person

Klaus Hain, DL8NAT
Jahrgang 1958,
Amateurfunkgenehmigung seit 1980
Gelernter Radio- und Fernsehtechnikermeister, beruflich in der Elektronikentwicklung tätig
Besondere Interessen: Bau von Geräten für den Amateurfunk
Weitere Hobbys: Radfahren

Anschrift:
Geiersgrund 4
95326 Kulmbach
dl8nat@darc.de

Grundüberlegungen für den Prozessoreinsatz

Da umfangreiche Rechenarbeit für den PIC zu erwarten war, wählte ich die maximale Taktfrequenz von 20 MHz. Die digitale Signalverarbeitung benötigt eine konstante Abtastrate. Die maximale Signalfrequenz beträgt 250,3 Hz. Um ohne ständige Manipulationen am Timer auszukommen, setzte ich die Abtastfrequenz auf 9766 Hz, entsprechend 20 MHz/2048. So ergeben sich ca. 39 Punkte pro Periode. Die Wiederholgenauigkeit liegt bei 200 ns. Das Signal wird wieder ausreichend hergestellt. Die Tabellen wurden mit Q-Basic erstellt und als ASCII in den Quellcode eingesetzt. Das Programm wurde mit MP-Lab V7.2 [5] in Assembler entwickelt.

Sendezweig als DDS-Generator

Ich entschied mich, zuerst den Sendezweig fertigzustellen. Hier wird das Prinzip des Direkten Digitalen Sythesi-

67,0Hz	71,9Hz	74,4Hz	77,0Hz	79,7Hz	82,5Hz
85,4Hz	88,5Hz	91,5Hz	94,8Hz	97,4Hz	100,0Hz
103,5Hz	107,2Hz	110,9Hz	114,8Hz	118,8Hz	123,0Hz
127,3Hz	131,8Hz	136,5Hz	141,3Hz	146,2Hz	151,4Hz
156,7Hz	162,2Hz	167,9Hz	173,8Hz	179,9Hz	186,2Hz
192,8Hz	203,5Hz	210,7Hz	218,1Hz	225,7Hz	233,6Hz
241,8Hz	250,3Hz				

Bild 1: Die in den Standard-Funkgeräten verwendeten CTCSS-Frequenzen [1]

zers angewandt. Eine 16 Bit breite Speicherzelle (Phasenakkumulator) stellt den Phasenwinkel von 0° bis 360° dar. Hier wird bei jeder Abtastung das Phasen-Inkrement dazu addiert. Es ist umso größer, je höher die Frequenz ist. Für jede der 38 CTCSS-Frequenzen ist ein Inkrement abgespeichert. Die „Sinustabelle" enthält 90° einer Sinuskurve vorberechnet (7 Bit breit).

Die Bits 7 bis 13 des Phasenakkumulators geben die Adresse der Sinustabelle (jeweils 90°) an. Bit 14 invertiert die Leserichtung (90...180°, 270°...360°) und Bit 15 legt das Vorzeichen (180°...360°) fest. Der Wert aus der „Sinustabelle" und das Vorzeichenbit werden dem Pulsbreitenmodulator (7+1 Bit) zugeführt. Nach einem Tiefpass erscheint die CTCSS-Frequenz am Ausgang (D/A-Wandler).

Kommunikation mit dem Hauptprozessor

Matthias hatte ein C-520 mit CTN-520-Orginalplatine aufgetrieben. Bewaffnet mit Speicheroszilloskop und einigen anderen Messgeräten, machten wir uns an die Analyse der Schnittstelle.

Wie aus **Bild 2** zu ersehen ist, handelt es sich um eine SPI-Schnittstelle (SO, /SCK). Sie bedient außer der CTCSS-Platine die PLLs des VHF- und UHF-Synthesizers. Mit einem Übernahmeimpuls (TEV, TEU) werden die letzten 8 Bits in den Speicher der angeschlossenen Bausteine geschrieben. Um bei weiteren Experimenten das Funkgerät nicht zu gefährden, baute ich in Fädeltechnik einen Schnittstellensimulator auf. Er stellt über die SPI-Schnittstelle die Verbindung zur CTCSS-Platine her. Für Testzwecke war auch ein Rückkanal vorhanden.

Goertzelfunktion als Rettung

Auf der Suche nach Lösungen für die Decodierung stieß ich auf die Goertzelfunktion [6, 7, 8, 9, 10, 11]. Es handelt sich dabei um einen Sonderfall einer FFT. Mit deren Hilfe lässt sich der relative Amplitudenanteil einer bekannten Frequenz herausrechnen. Man unterscheidet dabei Real- und Imaginäranteil.

Zum weiteren Verständnis nur Folgendes: Mit der optimierten Goertzelfunktion unterscheidet man nicht mehr zwischen Real- und Imaginäranteil. Nach einer Berechnungsschleife, die n mal durchlaufen wird (hier n = 256), erfolgt eine Auswertung. Um nur positive Werte zu erhalten, wird quadriert und dann die Wurzel gezogen. Da die Wurzelfunktion in Assembler nur aufwändig zu erzielen ist, beschränkt man sich oft auf die Quadrate: $(-a)^2 = +a^2$ und vergleicht diese miteinander.

Das Ganze hat einen Nachteil. Durch das Quadrat erhöht sich natürlich der Wertebereich gewaltig. Das lässt sich nur beherrschen, indem man die Rechenbreite auf 32 Bit oder mehr erhöht, was die Geschwindigkeit drosselt. Die Einführung einer Fließkomma-Arithmetik hat ähnliche Folgen und ist dabei noch sehr empfindlich gegen Störeinflüsse.

Ich implementierte die optimierte Goertzelfunktion in den Controller. Das Ergebnis wurde über die SPI-Schnittstelle nach jedem Durchgang am Display angezeigt. Beim Durchschalten der einzelnen Frequenzen war zwar rein intuitiv für die berechnete Frequenz ein Maximum feststellbar. Dies schwankte aber teilweise über mehrere Zehnerpotenzen und war mit einer festen Schaltschwelle nicht auszumachen. Es zeigten sich auch oft negative Werte, was auf einen Überlauf im Rechenwerk hindeutete.

Die „Technikerlösung"

Alle Mathematiker mögen mir den folgenden Lösungsweg verzeihen. Um die geheimnisvolle Goertzelfunktion zu ergründen, habe ich den Ausgang des Goertzelfilters (vor Auswertung) auf den D/A-Wandler gelegt. Jetzt war eine direkte Untersuchung mit dem Oszilloskop möglich. Die Überlauf- und Clipping-Effekte wurden sofort sichtbar.

Das Goertzelfilter stellt ein mitgekoppeltes System dar [9, 11]. Es entspricht in etwa einem idealen Oszillator (Kreisverstärkung 1, Phasenlage 360°). Ohne zugeführte Energie schwingt das System nicht. Ein Impuls stößt es auf der Resonanzfrequenz an, und es arbeitet als Sinusoszillator. Wird mehr Energie mit der Resonanzfrequenz zugeführt, erhöht sich kontinuierlich die Ausgangsamplitude. Liegt die Frequenz der zugeführten Energie neben der Resonanzfrequenz, wird dem System Energie entzogen, und die Amplitude geht zurück.

Etwas eigenwillig ist das Verhalten knapp neben der Resonanzfrequenz. Hier überlagern sich Eingangssignal und eigenes Signal („Schwebung"). Die Oszillogramme in **Bild 3, 4 und 5** zeigen die entsprechenden Signale. Die Amplitude des Signals mit Mittenfrequenz steigt kontinuierlich an. Das Filter wird nach n Abtastungen immer wieder auf null gesetzt.

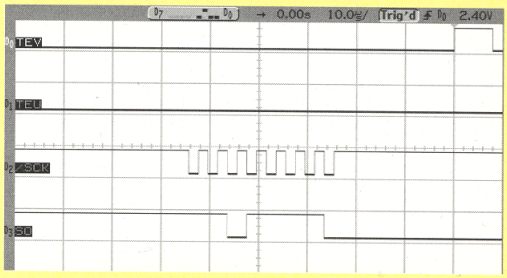


Bild 2: Hier sind die Steuersignale des Prozessors zu sehen, von oben: Übernahmeimpuls, Takt und Daten

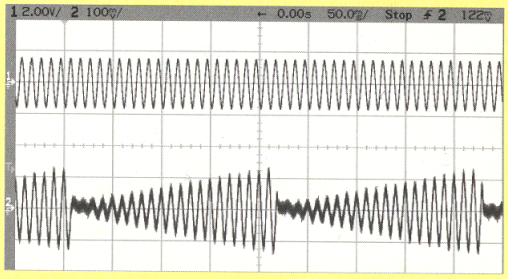


Bild 3: Dieses Bild zeigt das Goertzelfilter, abgestimmt auf die Mittenfrequenz. Die untere Kurve lässt einen kontinuierlichen Anstieg bis zum Zurücksetzen erkennen

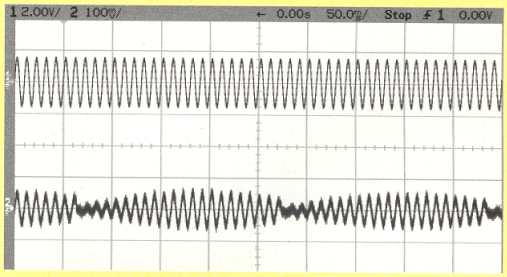


Bild 4: Das Filter ist hier auf eine CTCSS-Frequenz höher abgestimmt. Der lineare Anstieg ist verschwunden

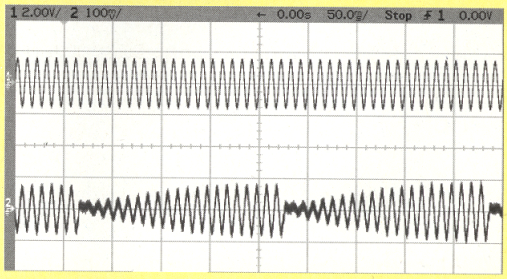


Bild 5: Das Filter ist auf eine CTCSS-Frequenz niedriger abgestimmt

Parallelverarbeitung ist schnell

Eine Filterkurve durchzurechnen, dauert ungefähr 210 ms. Da die Amplituden des Eingangssignals im beträchtlichen Maße schwanken können, ist ein Vergleich mit einem festen Schwellwert nicht möglich. Als Referenz können nur die Amplituden der Nachbarfrequenzen dienen. Deren

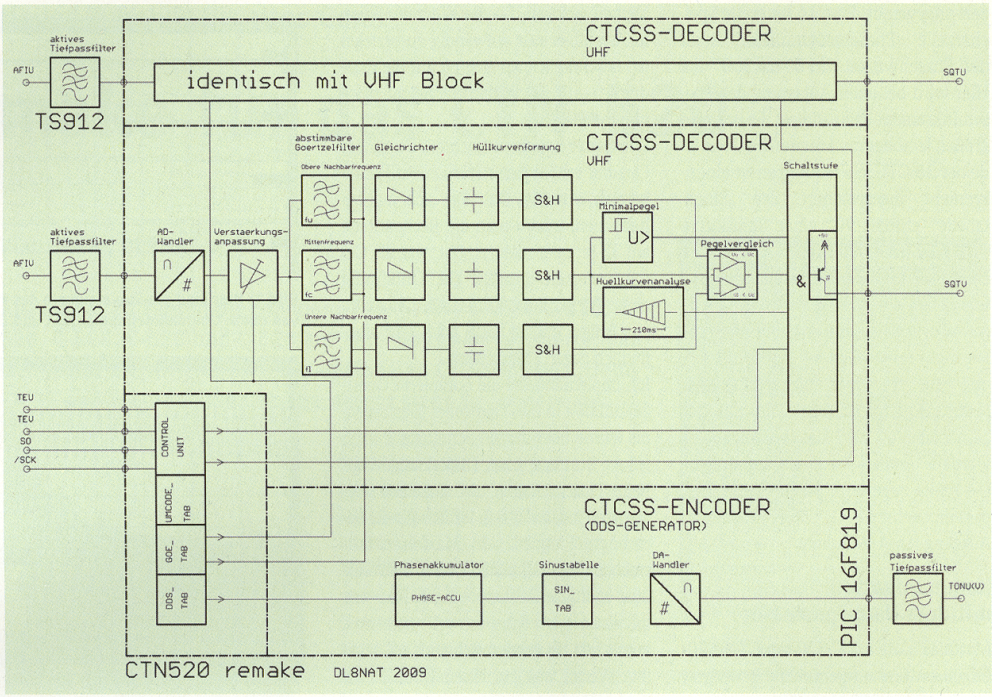


Bild 6: Das „Blockschaltbild" der Software. Die Blöcke für VHF und UHF sind identisch

Berechnung erfolgt deshalb parallel, und es stehen nach ca. 210 ms drei Ergebnisse bereit. Dazu nutzte ich eine Eigenheit des Prozessors aus, nämlich das Umschalten von Speicherbänken, das sonst oft sehr hinderlich ist. So aber gelang es, mit ein und demselben Programm drei Berechnungen quasi gleichzeitig vorzunehmen.

Die rechenaufwändige Gleichrichtung durch Quadrieren und anschließendem Radizieren habe ich dadurch vereinfacht, dass ich bei gesetztem Negativ-Bit den Spannungswert mit –1 multipliziere. Alle Werte kommen nun in einen Spitzenwertdetektor, der immer die maximale Amplitude speichert (Kondensator).

Auswertung der Signale

Die eigentliche Auswertung geschieht nun „grafisch". Die Kurve der Mittenfrequenz wird in acht Teile zerlegt und die jeweilige höchste Amplitude gespeichert. Von den beiden Nachbarfrequenzen wird der Amplitudenwert des letzten Achtels festgehalten. Die Frequenz ist erkannt, wenn die Werte der Mittenfrequenz kontinuierlich ansteigen und die beiden Nachbaramplituden im letzten Achtel kleiner sind als die der Mittenfrequenz und wenn ein Minimalpegel überschritten ist.

Der Chip und seine Software

Bild 6 versucht eine Blockdarstellung der Software. Alle Filterfunktionen greifen auf gemeinsame Tabellen zu. Die Goertzeltabelle enthält sowohl den Filterkoeffizienten als auch einen Korrekturfaktor, um Amplitudenunterschiede bei den verschiedenen CTCSS-Frequenzen sowie den Frequenzgang des Aliasingfilters auszugleichen.

Die DDS-Tabelle enthält die Inkremente für das Sendesignal. In der Umcodiertabelle werden die Kommandos vom Steuerprozessor an die Software angepasst. Nicht zuletzt sind 90° einer Sinuskurve abgespeichert. Damit lassen sich die restlichen 270° einfach berechnen. Die Kontrolllogik und die Schaltstufen stellen die Verbindung nach außen dar. Die Software wurde mit [5] entwickelt.

Die Hardware

Die meisten Funktionen sind in der Firmware implementiert. Daher bleibt die Hardware bescheiden.

Bild 7 zeigt die übersichtliche Schaltung und **Bild 8** die Bestückung.

Für jeden der beiden Empfangsbereiche gibt es ein Aliasingfilter. Dies verhindert ein Erkennen von Frequenzen, die durch

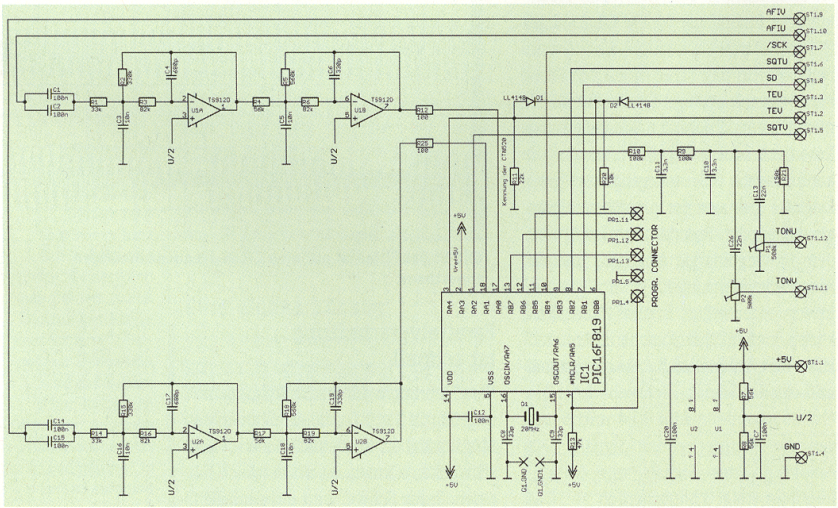


Bild 7: Der Schaltplan ist einfach, denn bis auf die aktiven Eingangsfilter und das passive Ausgangsfilter sind alle Funktionen im Mikrocontroller integriert

Mischung von Eingangs- und Abtastsignal entstehen. Es wurde mit [4] berechnet.
Das Tiefpassfilter für die PWM-Signale des DDS ist mit passiven Bauelementen diskret aufgebaut.
Die Transistoren der Originalplatine werden per Software simuliert. Die Ausgänge haben also die gleiche Funktion wie ein PNP-Transistor mit Open Collector.
Über einen Programmierstecker kann die Firmware auf die fertige Platine gespielt werden.
Der Schaltplan und das Layout wurden mit [12] entworfen. Da die Eagle-Light-Version eine kommerzielle Nutzung verbietet, hat DG1NMT mit seiner Vollversion die Weiterverarbeitung der Entwürfe übernommen, um sie für den Platinenhersteller aufzubereiten. Die Größe der Originalplatine war ausschlaggebend für diesen Entwurf. Es kamen daher nur SMT-Bauteile infrage.

Der Nachbau

Für „Selbstlöter" sind die Eagle-Files auf dem CQ DL-Server vorhanden. Die wenigen Durchkontaktierungen können mit dünnen Drähten erfolgen.
Nach reiflicher Überlegung habe ich mich entschlossen, die Firmware nicht offenzulegen. Jedoch erhält jeder interessierte Funkamateur von mir einen programmierten PIC zum Selbstkostenpreis plus Porto. Die Firmware ist natürlich lesegeschützt. Es genügt eine kurze E-Mail mit Angabe des Rufzeichens.

Literatur und Bezugsquellen

[1] Matthias Tafelmeyer, DG1NMT: www.ctcss.de
[2] CML: CRCSS Decoder/Encoder
[3] Consumer Microcircuits Ltd: Product Information FX335, November 1982
[4] www.captain.at/electronics/active-filter/
[5] www.microchip.com
[6] Stefan Jeschke: Die Detektion von DTMF-Signalen, Studienarbeit
[7] Kevin Banks, Embedded Systems Design, The Goertzel Algorithm
[8] H. Lenzen: CTCSS Decoder mit AT Tiny 2313 CPU, Oktober 2007
[9] Prof. Dr. Hans-Hellmuth Cuno, DL2CH: Praktische Elektronik, Digitale Signalverarbeitung, S. 10-1 bis 10-9
[10] Gene Small: Detecting CTCSS tones with the Goertzel algorithm, EE Times-India, April 2006
[11] Daniel Ch. von Grüningen: Digitale Signalverarbeitung, 4. Auflage, Hanser
[12] Eagle Light: www.cadsoft.de
[13] Fa. Standard, Schematic Diagram C-520

Für diejenigen, die den Lötkolben nicht schüren wollen oder den doch etwas gewöhnungsbedürftigen Umgang mit SMDs scheuen, hat Matthias einen Satz fertiger Platinen aufgelegt. Sie sind industriell bestückt, vorabgeglichen und bereits weltweit erfolgreich im Einsatz.
Der Einbau erfolgt nach Anweisung des Geräteherstellers. DG1NMT gibt auf seinen Webseiten wertvolle Tipps und Hinweise für Standard-Nutzer. Die Hubregler für 2 m und 70 cm sind auch beim Nachbau vorhanden.

Der Vergleich

Der Nachbau „CTN520 remake" braucht den Vergleich mit dem Original nicht zu scheuen, siehe **Bild 9**. Ein auf den ersten Blick nicht zu erkennender Unterschied soll aber nicht verschwiegen werden. Beim Original ist es theoretisch möglich, in beiden Frequenzbereichen mit unterschiedlichen CTCSS-Frequenzen gleichzeitig zu senden. Dies ist aber vom Funkgerätehersteller nicht vorgesehen. Deshalb schränkt die gemeinsame Erzeugung der UHF- und VHF-CTCSS-Frequenz per PWM die Funktionalität nicht ein.
Die Schaltung funktioniert sehr sicher und bietet sich deshalb zum nachträglichen Einbau an.

Ausblick

CTCSS wird auch in DL immer mehr zum Thema. Bausteine zum Decodieren sind schlecht verfügbar. Einige Funkamateure möchten aber ihre alten Geräte nachrüsten. Um das Know-how über Decodierung und Codierung von CTCSS-Frequenzen, das mit dieser Entwicklung erworben wurde, nicht ungenutzt zu lassen, sind schon ein paar neue Projekte in Arbeit. So dürfte eine Miniaturplatine (nur Encoder) zum Einbau in fast jedes Funkgerät bei Erscheinen der Beitrags erhältlich sein. Geplant sind ein De- und Encoder für Heimstationen mit sehr einfacher Einstellung der CTCSS-Frequenz sowie ein Modell für Amateurfunkrelais mit Offenlegung der Schnittstelle zur Einbindung in die Relais-Firmware.
Zum Schluss bedanke ich mich bei Matthias, DG1NMT, der mich bei der Entwicklung mit endlosen Testreihen und bei der Beschaffung von seltenen Bauteilen tatkräftig unterstützt hat. CQ DL

Stückliste

Bauteil	Wert	Bauform
C1	100 n	C0805
C2	100 n	C0805
C3	10 n	C0805
C4	680 p	C0805
C5	10 n	C0805
C6	330 p	C0805
C7	100 n	C0805
C8	33 p	C0805
C9	33 p	C0805
C10	3,3 n	C0805
C11	3,3 n	C0805
C12	100 n	C0805
C13	22 n	C0805
C14	100 n	C0805
C15	100 n	C0805
C16	10 n	C0805
C17	680 p	C0805
C18	10 n	C0805
C19	330 p	C0805
C20	100 n	C0805
C26	22 n	C0805
D1	LL4148	D_LL4148
D2	LL4148	D_LL4148
IC1	PIC16F819	PIC16F84SO SO18
P1	500 k	C4312
P2	500 k	C4312
Q1	20 MHz	MJCRYSTALS
R1	33 k	R0805

Bauteil	Wert	Bauform
R2	330 k	R0805
R3	82 k	R0805
R4	56 k	R0805
R5	560 k	R0805
R6	82 k	R0805
R7	56 k	R0805
R8	56 k	R0805
R9	100 k	R0805
R10	100 k	R0805
R11	22 k	R0805
R12	100	R0805
R13	47 k	R0805
R14	33 k	R0805
R15	330 k	R0805
R16	82 k	R0805
R17	56 k	R0805
R18	560 k	R0805
R19	82 k	R0805
R20	10 k	R0805
R21	150 k	R0805
R25	100	R0805
ST1	Wannenleiste 1,5 mm + Kontakte (DIGIKEY)	
STxx	Litze 0,1 mm	ca. 0,6 m
U1	TS912D	LM358SMD SO8
U2	TS912D	LM358SMD SO8
LP	Leiterplatte	1/12 Europakarte

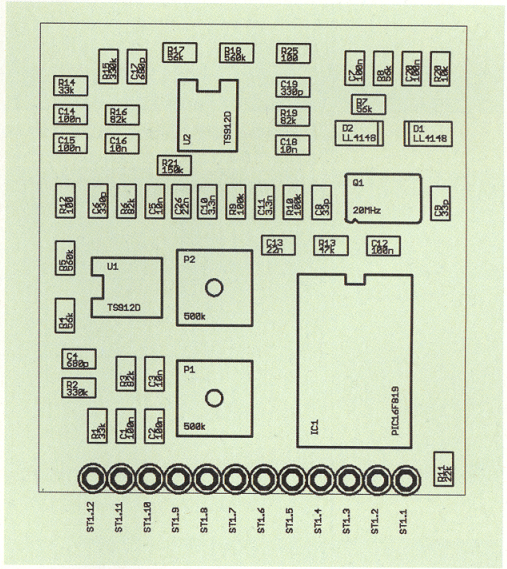


Bild 8: Der Bestückungsplan

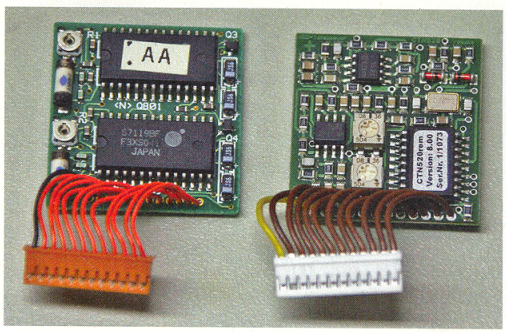


Bild 9: Das Original CTN-520 (links) und der Nachbau (rechts). Die Buchsenleiste hat das nicht gerade übliche Rastermaß 1,5 mm