De Bug Butcher

door PA0WV Wim

Inleiding

Een bug (vibroplex) geeft door een trillende pendel bij het naar rechts duwen met de duim van de rechterhand een serie punten. Duw je met je wijsvinger naar links, dan krijg je een streep zolang je duwt. Er zijn ook bugs voor linkshandigen, dan gaat dat net andersom, omdat links je duim rechts zit.

Het kan aardig zijn met wat elektronica die duw niet per streep te hoeven maken. Een cijfer 0, vaak voorkomend als je een pa0 call hebt, vereist immers 5 duwacties. Als je een langere duw in strepen kunt verdelen, zoals de bug dat zelf met punten doet, en die dan keurig afgepast in de juiste mark-space verhouding zou genereren zolang je de paddle met de wijsvinger ingeduwd houdt, dan heb je daar een hoop gemak van. Het wordt dan ook makkelijker om een bug en een keyer afwisselend te gebruiken, omdat ze dan dezelfde bediening van de paddle vereisen.

Dat doen we in dit artikel en de snijactie van de superlange duw naar links op de paddle, die in keurig afgepaste strepen wordt verdeeld, is een keurslageractie; vandaar de naam van het apparaatje 'Bug Butcher' die ik ervoor heb bedacht.

Ontwerp

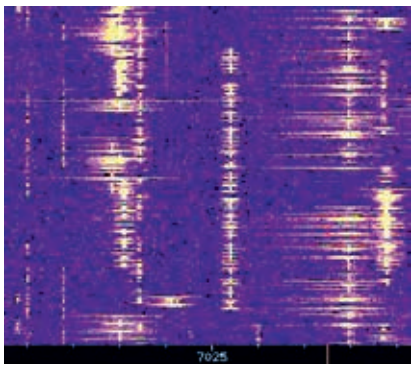
De streeplengte moet drie maal de puntlengte worden. De puntlengte wordt bepaald door de plaats van de gewichtjes op de pendel en is dus variabel. De streeplengte moet automatisch in de verhouding 3 op 1 volgen. Je moet dus de puntlengte voortdurend meten om dat te kunnen realiseren.

Blijkt de gesloten contacttijdsduur langer dan 2 punten, dan wordt de streep afgemaakt, inclusief de bijbehorende spatie. Is het bugcontact daarna dan nog steeds gesloten dan volgt een volgende volledige streep.

Het heeft een negatieve invloed op je bug-fist als je verplicht wordt je laatste streep korter dan 3 lang te houden om een volgende ongewenste streep te voorkomen. Dit wordt voorkomen door het testmoment, dat bepaalt of er nog een streep moet komen, op het begintijdstip van de volgende streep te leggen.

Bounce

Een mechanisch contact geeft bounce of contactdenderen. Dat wil zeggen dat zowel bij maken als verbreken van het contact er een aantal snelle maak- en verbreekacties bijkomen. Als je daar een eenvoudige zender mee sleutelt heb je kans op flinke bandbreedte ten gevolge van sleutelklik. Bij een dure jappenbak trouwens ook, zo blijkt als ik het spectrum bekijk dat de Websdr ontvanger van de TU-Twente toont op een willekeurig weekeinde dus gedurende een of andere contest.



Voor de werking van de Bug Butcher zou dat trouwens helemaal rampzalig zijn, want die ziet het snelle schakelen gedurende bouncing als zeer korte punten en maakt de streep dan driemaal dat bedrag lang.

Dat kan vermeden worden als we de uitgang van de Bug Butcher, die de zender sleutelt, inschakelen op de eerste maak van het bouncende sleutelcontact, snelle onderbrekingen daarna niet op reageren, en de uitgang uitzetten op de eerste verbreekactie van het inmiddels langer gesloten contact. Leading en trailing edge worden dan bouncevrij, zonder de tekenduur aan te tasten. Theoretisch hebben we dan wel een snelheidslimiet; de punten kunnen immers niet

Le Bug Butcher

par PA0WV – traduit par ON5WF

Introduction

Un vibroplex (bug) donne, au moyen d'une lame vibrante, une série de points lors d'une pression exercée vers la droite par le pouce de la main droite. Si l'on appuie vers la gauche avec l'index, on obtient alors un trait qui dure tant que l'on appuie. Il existe aussi des vibroplex pour gauchers ; le fonctionnement est alors inversé puisque chez un gaucher, le pouce se trouve à droite.

Avec un peu d'électronique, il est possible d'éviter de devoir effectuer un appui pour chaque trait. Un chiffre 0, fréquent dans le cas d'un call pa0, par exemple, impose en effet 5 actions d'appui. Si l'on peut décomposer un appui long en une série de traits, tout comme le bug le fait lui-même avec les points, et que ces traits sont correctement séparés, on a alors gagné beaucoup en confort. Il est alors aussi plus facile d'utiliser indifféremment un bug ou un keyer, car ils fonctionnent dans ce cas de la même façon.

C'est ce que nous allons faire dans cet article. L'action de découpe du très long appui vers la gauche sur le levier qui produit une série de traits correctement espacés, peut être comparée à une découpe de boucherie, d'où le nom de 'Bug Butcher' choisi pour cet appareil.

Conception

La durée d'un trait doit être égale à trois fois la durée d'un point. La durée d'un point est déterminée par la position de la masselotte sur la lame vibrante du vibroplex ; elle est donc variable. La durée d'un trait doit rester automatiquement dans le rapport 3/1 par rapport au point. Pour arriver à ce résultat, il faut mesurer continuellement la durée du point.

Si la durée du contact est supérieure à celle de deux points, le trait est alors terminé, l'espacement correspondant compris. Si le contact est alors ensuite encore toujours fermé, il s'en suit un nouveau trait complet.

Cela a une influence négative sur le poignet si l'on doit maintenir la durée du dernier contact inférieure à 3 points afin d'éviter la formation d'un nouveau trait suivant. On peut éviter cela en plaçant au tout début du trait suivant le test qui détermine si un trait doit encore être formé.

Rebond

Un contact mécanique donne lieu à des rebonds. Cela signifie que, lors de la fermeture ou de l'ouverture du contact, il se produit un certain nombre de fermetures et ruptures rapides du contact. Si l'on commande de cette façon un émetteur simple, ces "clicks" de manipulation vont augmenter la bande de fréquences occupée par l'émetteur. Avec un coûteux transceiver moderne aussi d'ailleurs, ce que montre le spectre observé sur le récepteur Websdr de TU-Twente lors d'un WE quelconque, donc pendant l'un ou l'autre contest.

Pour le fonctionnement du Bug Butcher, cela serait d'ailleurs complètement catastrophique. En effet, ces fermetures et ouvertures rapides dues aux rebonds sont vues comme des points très courts, ce qui rend alors la durée du trait également plus courte.

Cela peut être évité si nous faisons en sorte que la sortie du Bug Butcher qui commande l'émetteur se ferme à la première fermeture du contact et ne réagisse plus ensuite aux rebonds suivants et que la sortie s'ouvre lors de la première rupture du contact à la fin du trait. Les flancs montants et descendants sont alors exempts de rebonds et la durée du trait n'est plus

korter worden dan de veronderstelde maximaal optredende bouncetijd. maar dat konden ze toch al niet.

Uitwerking van het ontwerp

Gebruik wordt gemaakt van de hier in een laatje op voorraad liggende microcontroller AT89S8253 die ik ook in andere ontwerpjes heb gebruikt, welke op mijn website http://pa0wv.home.xs4all.nl/zelfbouw.html te vinden zijn.

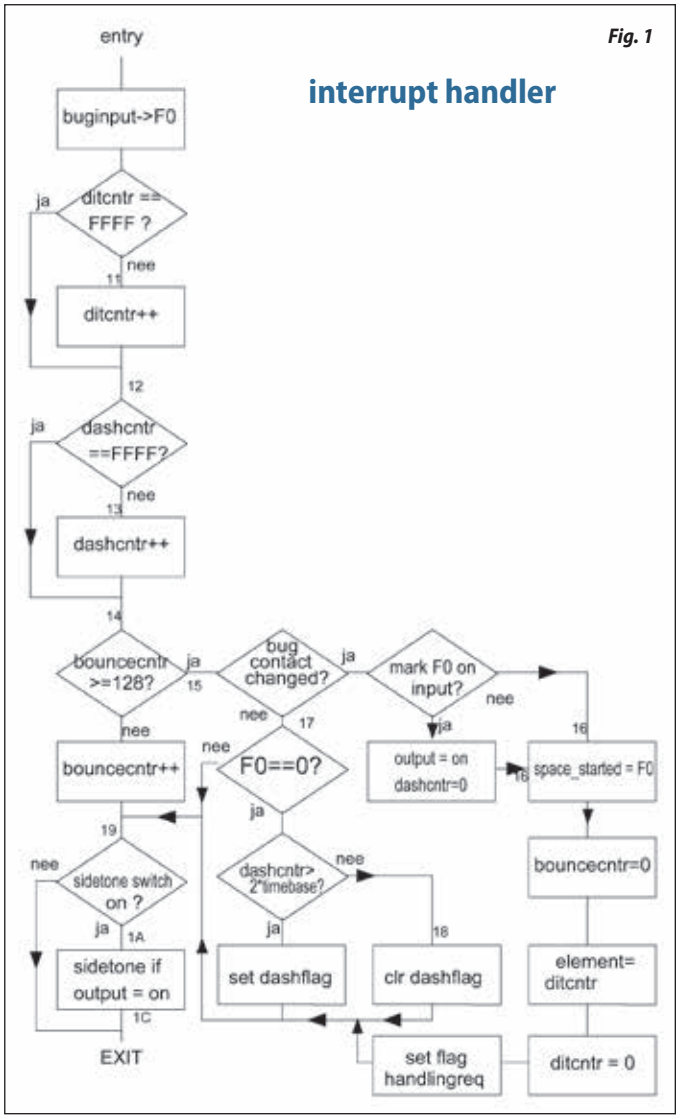
Ik programmeer die in assembler. Assembler is een programmeertaal die per type chip verschilt, niet handig dus, maar het voordeel is dat je precies weet wat er gebeurt en de chips dus optimaal kunt gebruiken.

Nu kun je een programma ook omzetten in een andere chip, een Arduino, een Raspberry Pi of zoiets, indien je de flow chart beschikbaar hebt. Die flow charts teken ik er dus bij in dit artikel, en ik leg de werking uit, zodat je zelf, indien gewenst, kunt experimenteren met een andere processor en een andere programmeertaal. Per slot van rekening hebben we onze machtiging of registratie verkregen om te experimenteren en onszelf te ontwikkelen, dat wordt maar al te vaak vergeten door 5nn tu contesters en dx-jagers met jappenbakken.

De werking

Een timer veroorzaakt ruim 7800 keer per seconde een interrupt iedere keer als hij van 255 naar 0 flipt. Dat is namelijk de kristalfrequentie/6/256 keer per seconde. De afhandelingsroutine van die interrupts (zie **figuur 1**) neemt de stand van de bug op, mark of space, verhoogt een 16 bits dit-duurteller 'ditcntr' zolang die niet op zijn maximum 65535 is beland na ruim 8 seconde en verhoogt tevens een een-byte brede bouncetijdteller, mits die laatstgenoemde beneden de maximale bouncetijdcount staat en doet

Fig. 1



affectée. Théoriquement, il y a alors bien une limite à la vitesse. Les points ne peuvent en effet pas avoir une durée plus courte que celle d'un rebond ; mais pratiquement, cela n'est quand même pas possible.

Mise en oeuvre

J'ai utilisé un microcontrôleur AT89S8253 que j'avais en réserve et que j'avais déjà utilisé pour d'autres projets ; voir à ce sujet mon site web http://pa0wv.home.xs4all.nl/zelfbouw.html.

Je le programme en assembleur. L'assembleur est un langage de programmation qui varie d'un contrôleur à l'autre. Pas pratique donc, mais l'avantage est que l'on sait exactement ce qui se produit et que l'on peut donc utiliser le contrôleur de façon optimale.

Maintenant, on peut adapter le programme pour un autre contrôleur, comme un Arduino, un Raspberry Pi ou un autre du même genre, pour autant que l'on dispose de l'organigramme. Celui-ci est donné et expliqué dans cet article, de façon à ce que, si vous le souhaitez, vous puissiez expérimenter avec un autre processeur et un autre langage de programmation. En fin de compte, nous avons l'autorisation d'expérimenter et développer notre propre matériel ; cela est trop souvent oublié par les "5nn tu" des contesters et chasseurs de DX avec leurs "machines automatiques".

Le fonctionnement

Un timer0 produit au moins 7800 fois par seconde une interruption chaque fois qu'il retombe de 255 à 0. Il s'agit en fait de la fréquence du quartz/6/256. La routine de traitement de ces interruptions (voir la **figure 1**) enregistre l'état du bug, mark ou space, incrémente un compteur de 16 bits de durée des "dits", "ditcntr", tant que celui-ci n'a pas atteint son maximum de 65535. Ce qui prend 8 bonnes secondes. Il augmente en même temps un compteur de temps de rebond, de 1 byte de largeur, pour autant que ce dernier se trouve en dessous du temps maximum de comptage de rebond. Ensuite, à part produire une tonalité si la sortie se trouve sur mark, il ne fait rien de plus. Si le compteur de durée de rebond a atteint son maximum autorisé, ce dernier n'est alors plus incrémenté. Dans ce cas, pour chaque échantillonnage, on regarde si le contact de la clé a été modifié de l'état bas à l'état haut (ouvert) ou l'inverse (fermé).
Si c'est le cas, le compteur de temps de rebond est mis à 0 et la valeur du compteur de durée de "dit" est stockée dans "element" pour être comparée dans le programme principal "main", avec les 8 dernières valeurs de durée de dits conservées. La plus petite de ces valeurs est utilisée comme base de temps pour déterminer la longueur des traits. Ces 8 valeurs peuvent être des marks ou des spaces, car la durée de chaque élément est déterminée par ditcntr. Lors d'un changement de contact du bug, la routine de traitement resette immédiatement le compteur de durée de dit car un nouveau mark ou space a alors commencé. Par ailleurs, un drapeau nommé space_started est mis à l'état haut lorsque le contact du bug est passé de "fermé" à "ouvert". Autrement, ce drapeau est mis à l'état bas. Cependant, si la clé n'a pas changé de position, ce qui est le plus souvent le cas si on vérifie cela 7800 fois par seconde, il ne se passe rien si un espace est en cours. S'il y a cependant un mark en cours, on vérifie alors si, entre temps, ce mark a une durée supérieure à deux fois la base de temps. Si oui, un drapeau de "dash" est alors mis à l'état haut, car dans ce cas, un dash doit être terminé, même si le contact du bug est relâché trop tôt. A chaque modification de la clé, le ditcntr est remis à zéro pour déterminer la longueur du prochain mark ou space. Dans le programme principal, un compteur (dash counter) vérifie en permanence si le trait est terminé et si la fin de l'espace qui le suit est atteinte. Lorsque c'est le cas, ce compteur est remis à zéro. Après chaque signe, un drapeau "handlingreq" est mis à l'état haut dans la routine d'interruption. Le programme principal conserve ce drapeau et veille à ce que la dernière mesure de base de temps dans "element" soit prête et puisse être traitée.

Si la routine de traitement constate qu'un mark a commencé, la sortie qui commande l'émetteur est mise à l'état bas.

Le programme principal (**figure 2**) conserve ce drapeau et le remet à zéro lorsqu'il est à l'état haut. Le programme principal prend alors la longueur du signe mesurée dans "element" (un mark ou un espace) et la stocke dans le buffer à 8 positions, à la place de l'élément le plus ancien. De cette façon, on dispose toujours de la durée des 8 derniers éléments. Ensuite, on

dan verder niks, buiten het maken van een sidetone als de output op mark staat. Staat de bouncetijdteller echter wel reeds op zijn toegelaten maximum dan wordt die niet verder verhoogd. In dat geval wordt per sample (bemonstering) gekeken of het sleutelcontact gewijzigd is van laag naar hoog (open) of omgekeerd (gesloten).
Is dat het geval dan wordt de bouncetijdteller op 0 gezet en de ditduurtellerwaarde bewaard in 'element' om in het hoofdprogramma 'main' te vergelijken met de laatste 8 bewaarde ditduurtellerwaarden, en de kortste van die verzameling wordt als tijdbasis gebruikt om de lengte van de strepen te bepalen. Die 8 waarden kunnen marks of spaces zijn, want van elk tekenelement wordt met ditcntr de lengte bepaald. De ditduurteller in de afhandelingsroutine wordt bij bugcontactwijziging dan onmiddellijk daarna gereset omdat er een nieuwe mark of space is begonnen. Tevens wordt een vlag genaamd space_started geset als het bugcontact in stond en naar uit is gegaan, andersom wordt die vlag gereset. Is de sleutel evenwel niet van positie gewijzigd, meestal dus als je 7800 keer per seconde kijkt, dan gebeurt er ingeval een space bezig is verder niets. Is er echter een mark aan de gang, dan wordt gekeken of de mark inmiddels langer dan 2 keer de tijdbasis duurt. Zo ja, dan wordt een dash flag gezet, omdat in dat geval een dash moet worden afgemaakt, ook als het bugcontact te vroeg wordt losgelaten. De ditcntr wordt bij elke sleutelwijziging weer op 0 gereset om de lengte van de volgende mark of space te meten, er loopt echter ook een dash counter die doorloopt, om in main te kunnen bekijken of de volledige streeplengte is bereikt en de bijbehorende spatie erachteraan. Daarna wordt in main die counter gereset. Na elk tekenelement wordt in de interruptroutine een vlag geset 'handlingreq', het programma 'main' bewaakt die vlag en ziet er aan dat het laatst gemeten tijdsduur in 'element' klaar staat en behandeld kan worden.

Als de afhandelingroutine vaststelt dat er een mark is begonnen, wordt de output, die de zender bedient, laag gemaakt.

Het programma main (**figuur 2**) bewaakt die vlag, als hij geset is wordt hij door main gereset. Main pakt dan de gemeten elementlengte uit 'element', dat kan een mark of space zijn, zet die in de 8 positiebuffer op de plek van het oudste element, zodat altijd de tijdsduur van de laatste 8 elementen beschikbaar staan. Vervolgens wordt het kortste van die 8 bepaald en opgeborgen in 'timebase'. Tevens wordt 2, 3 en vier keer dat bedrag berekend en in variabelen opgeborgen. Nodig om de strepen te construeren bij aanhoudend indrukken van de bugpaddle in de streeprichting. Na 8 verzonden tekenelementen, waartoe zowel marks als spaces worden gerekend, is die dan uit de buffer verdwenen en doet niet meer mee, dat is van belang als je de bug op een lagere snelheid zet met het pendelgewicht.

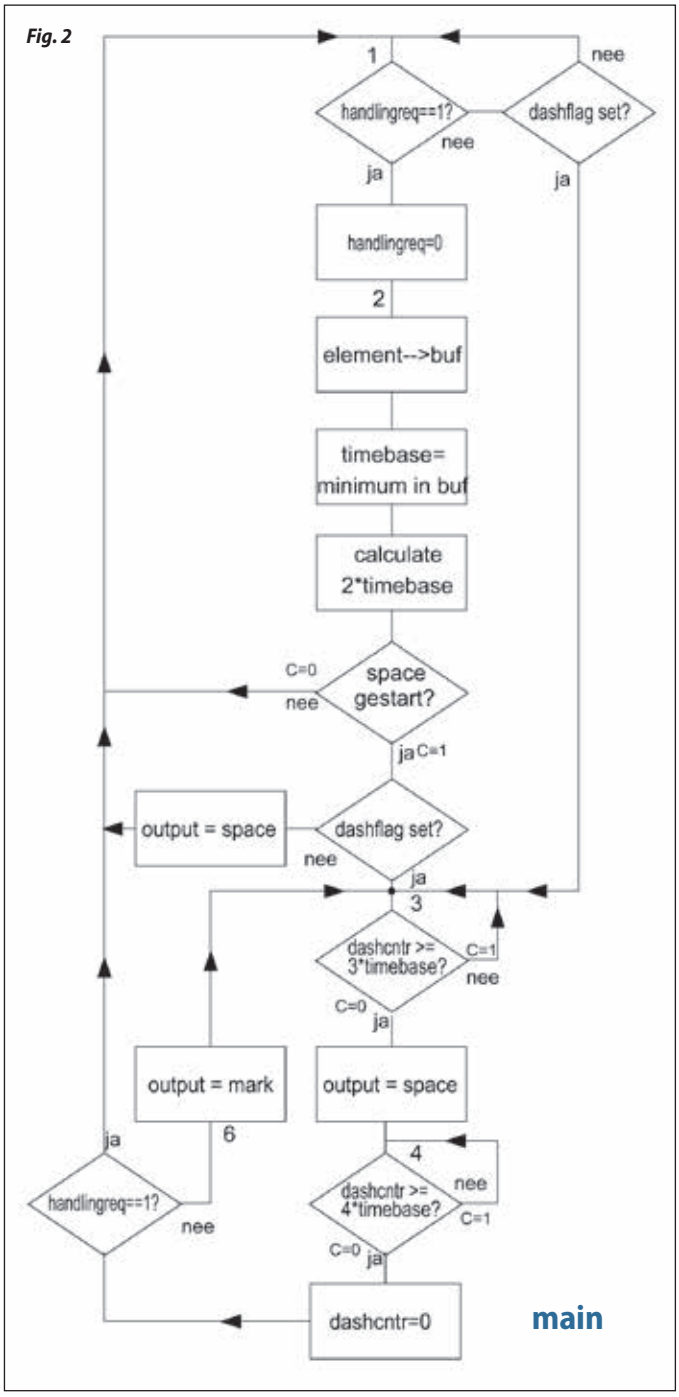
Een 'dit' is in dit verhaal gedefinieerd als de kortst voorkomende mark of space. Als we de bug tot 50 wpm willen kunnen gebruiken, is de minimale ditduur 24 ms. Daaruit volgt voor de maximaal toegelaten bouncetijd dat de bouncetijdteller nooit boven 187 kan komen bij een bouncetijd van 24 ms. 128 is een makkelijke te behandelen waarde om als toegelaten maximum te nemen wat wel ruim voldoende zal zijn. Bij een test, waarover verderop meer, blijkt dat vermoeden correct, alleen als je gaat seinen met twee krokodilklemmen tegen elkaar tikken gaat het fout, een gewone sleutel en bug geeft geen enkel probleem.

Nu treedt er een moeilijkheid op. Als je de streep indrukt en dat blijft doen, wordt die mark niet afgemaakt, dus komt er geen handlingreq uit de interrupt routine. Daarom draait in de interruptroutine ook een dashcounter mee die daar nooit gereset wordt, en ingeval er een mark wordt geseind, wordt daar met die dashcntr bijgehouden of die mark inmiddels langer dan 2 dits duurt dat is twee maal de gemeten en vastgestelde tijdbasis, en zo ja dan wordt in de interruptroutine een dashflag geset. In main wordt niet alleen gewacht op de handlingreq vlag voor het geval er een element klaar is, maar wordt ook de dashvlag bewaakt, en zodra die geset is in de interruptroutine doordat de paddle langer dan 2 dits is ingedrukt, wordt die dah afgemaakt en gevolgd door een bijbehorende spatie. Daarna wordt gekeken of de paddle nog steeds in staat en zo ja, dan wordt de dashcntr gereset op 0 en volgt tevens een volgende volledige streep.

Ingeval van een overgang van space naar mark, wordt in de interruptroutine de output op mark gezet. Terugzetten op space gebeurt in main.

Is het lopende teken een mark en langer dan twee dits, dan wordt de output in main pas teruggezet op space als de dashcntr in de interruptroutine drie

Fig. 2



calcule la plus courte de ces 8 durées et on la stocke dans "timebase". Cette valeur est aussi calculée 2, 3 et 4 fois et stockée dans des variables. Cela est nécessaire pour fabriquer les traits lors du maintien du contact de la clé du côté des traits. Après l'envoi de 8 signes, au moyen desquels aussi bien les marks que les spaces sont calculés, la valeur est supprimée du buffer et n'est plus utilisée. Cela est important pour le cas où l'on modifie la vitesse de vibration de la palette du bug.

Un "dit" est ici défini comme le plus court mark ou space à venir. Si nous voulons pouvoir utiliser le bug jusqu'à 50 wpm, la durée minimum d'un dit est de 24 ms. Il en résulte que, pour le temps de rebond maximum autorisé, le compteur de temps de rebond ne doit jamais dépasser 187 dans le cas d'un temps de rebond de 24 ms. 128 est une valeur facile à traiter comme maximum autorisé, ce qui sera bien largement assez. Lors d'un test qui sera traité plus en détail plus loin, il semble que cela soit correct, sauf si l'on fait du morse avec deux pinces crocodile, sinon, une simple clé ou un bug ne posent aucun problème.

Maintenant, il y a un problème. Si on maintient le contact fermé pour un trait, ce trait n'est pas terminé, la routine d'interruption ne fournit donc pas de handlingreq. Pour cette raison, dans la routine d'interruption, un dashcounter tourne sans jamais être remis à zéro. Dans le cas où un mark

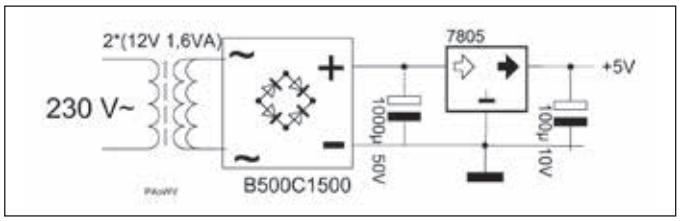


Fig. 3. Schema van de voeding / *Schéma de l'alimentation*

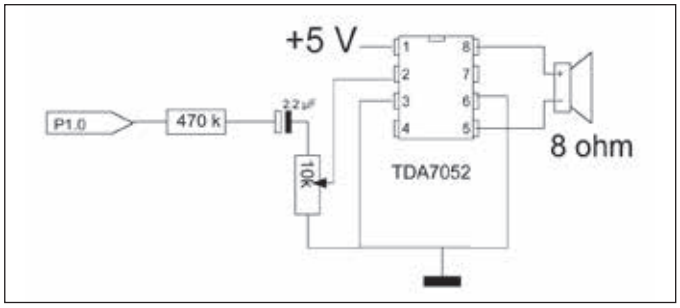


Fig. 5. Audioversterker / *Ampli audio*

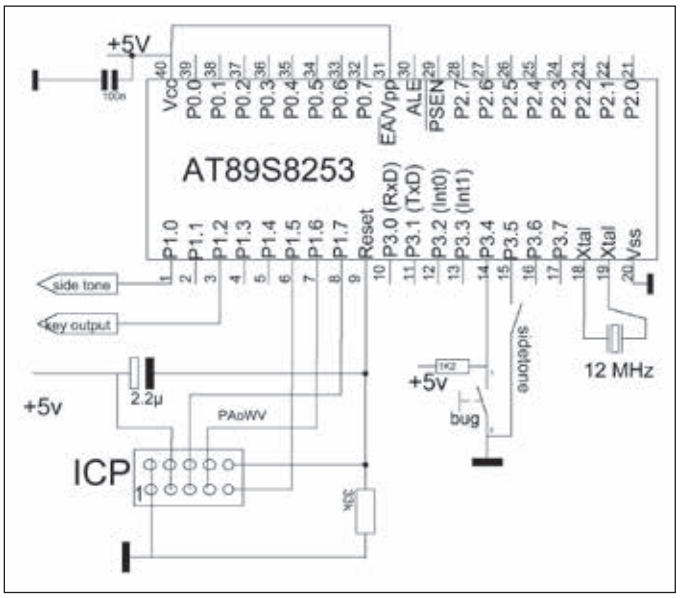


Fig. 4. Controller en periferie / *Contrôleur et périphériques*

dittijden heeft bereikt. Als die teller drie dits bereikt, een volle streep dus, wordt de output afgeschakeld en bij het bereiken van het einde van de vierde dit wordt gekeken of de input laag is (streep van de bug nog steeds ingeduwd). Zo ja, dan wordt de dashteller gereset en dit verhaal herhaald om de volgende streep te maken. Dit gaat door zolang de wijsvinger de paddle op de strepenstand houdt.

In de interruptafhandelingsroutine wordt ook gekeken of de output aan is, zo ja dan wordt een sidetone in de vorm van een blokgolf gegenereerd als pin P3.5, de sidetone schakelaar, hoog is, anders niet. Een versterker stuurt een luidspreker aan die de sidetone laat horen. Volume is instelbaar met een potmeter, die ik als trimpot op de print heb gemonteerd.

Test van de apparatuur

Het apparaat is gebouwd voor gebruik met een bug. Je kunt hem ook testen met een K1EL keyer (http:\\pa0wv.home.xs4all.nl\pdfbestanden\K1ELbug.pdf) door die in mode 'bug' te zetten. Dat gaat als volgt: inschakelen netspanning, wachten op R, cmd knop indrukken, wacht op R, geef een K met de paddle, de keyer meldt dan de ingestelde mode. bijvoorbeeld KB voor Iambic B mode. Raak de streeppaddle herhaaldelijk aan om door de modes te scrollen. Als je de V hoort weer op de cmd knop drukken, dan staat hij in bug mode. De keyer meldt succes weer met een R. Punten dan dus automatisch en streep zolang je die paddle indrukt.

Alles werkt niet direct zoals gewenst. Om te debuggen heb ik er een LCDisplay aangehangen via een 16 pins boxed header op port 2 (**figuur 6**). Routines die ik heb gebruikt voor het opsporen van fouten staan onder diagnostics in het programma opgenomen.

Om meer te weten te komen over de bounces die optreden bij sleutelwisselingen heb ik een alternatieve interruptroutine geschreven, als diagnostic. Bij het geven van een reeks dits, wordt bij elke maak en breek van het contact het aantal bounces gemeten en de langste bounceduur. Na 7 seconden krijg je dan het maximum van de bouncetijdsduren en het maximum van het aantal bounces per kontaktwisseling te zien op de LCD.

Gewone sleutels geven 1 of 2 bounces, maar ga je sleutelen door twee draadjes tegenelkaar te tikken dan krijg je wel hoge getallen. Die gekozen 128 van de bouncetijdsduur (16 ms) is dus inderdaad aan de zeer veilige kant. De flowchart van de alternatieve interruptroutine is bijgetekend in **figuur 7**.

Nu het programma klaar is en goed werkt, is uiteraard die LCD en de aansluiting op de processorport P2 niet meer nodig. De

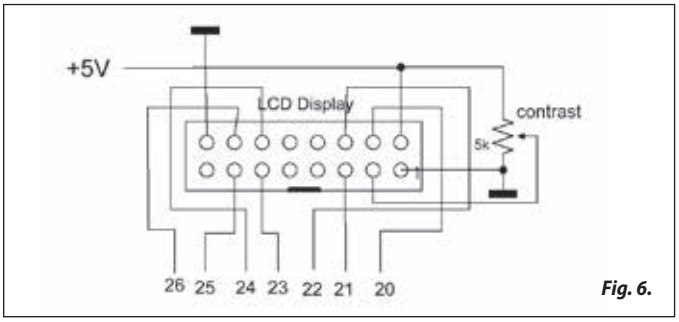


Fig. 6.

est envoyé, dashcntr permet de voir si le mark dure plus longtemps que deux dits, c'est-à-dire deux fois la valeur de la base de temps ; et si oui, un dashflag est mis à l'état haut dans la routine d'interruption.
Dans le programme principal, on ne surveille pas seulement le drapeau handlingreq pour le cas où un élément est prêt, mais aussi le dashflag. Et dès que celui-ci est mis à l'état haut dans la routine d'interruption parce que le contact est maintenu plus longtemps que deux dits, ce dah est terminé et suivi de l'espace correspondant. Ensuite, on vérifie si le contact de la clé est toujours fermé et, si oui, alors dashcntr est remis à zéro et il s'ensuit un nouveau trait complet.

Dans le cas d'un passage de space à mark, la sortie est positionnée sur mark dans la routine d'interruption. Le repositionnement sur space est effectué dans le programme principal.

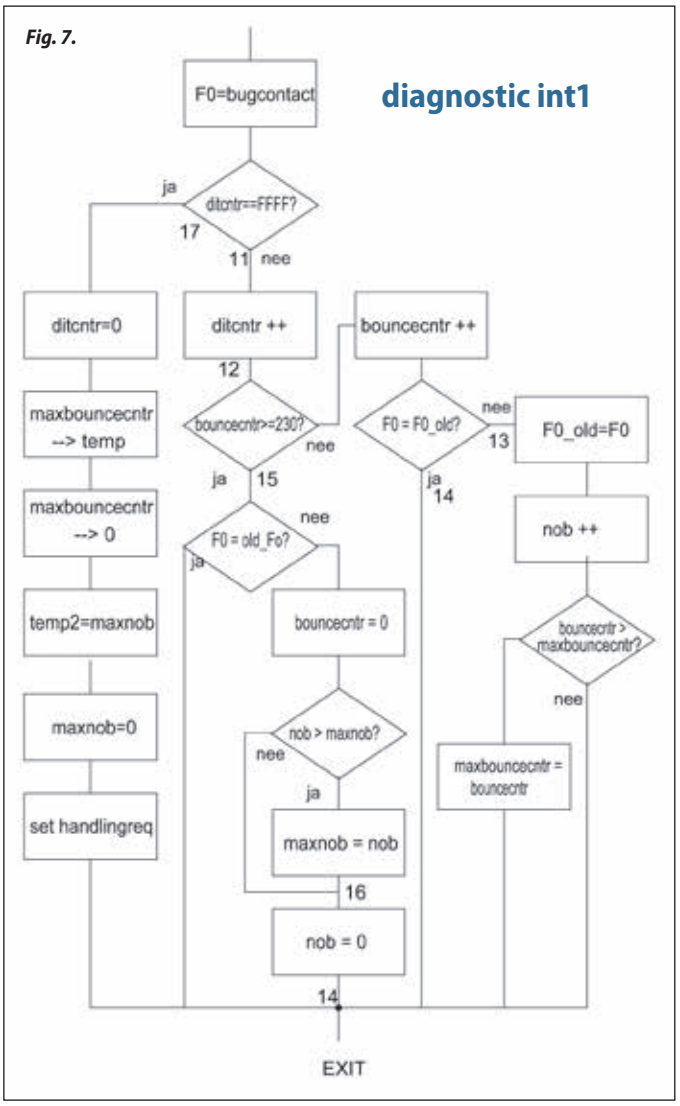
Si le signe en cours est un dah plus long que deux dits, la sortie est alors remise sur space dans le programme principal, seulement lorsque dashcntr, dans la routine d'interruption, a atteint trois temps élémentaires (trois dits). Lorsque ce compteur atteint trois dits, un trait complet donc, la sortie est déconnectée et, à la fin du 4ème dit, on vérifie si l'entrée est à l'état bas (contact "trait" du bug encore toujours fermé). Si oui, le compteur dash est alors remis à zéro et ce processus se répète pour produire le trait suivant, cela aussi longtemps que le contact trait reste fermé.

Dans la routine de traitement de l'interruption, on regarde aussi si la sortie est activée. Si c'est le cas, un signal carré est généré si la broche P3.5. L'interrupteur de tonalité est alors à l'état haut. Cette tonalité est amplifiée pour pouvoir être audible via un haut parleur. Le volume de l'ampli est réglable au moyen d'un trimpot monté sur le circuit imprimé.

Test de l'appareil

L'appareil est conçu pour être utilisé avec un bug. On peut aussi le tester avec un keyer K1EL (http:\\pa0wv.home.xs4all.nl\pdfbestanden\K1ELbug.pdf) en plaçant celui-ci en mode "bug". Cela se fait comme suit : mettre l'appareil sous tension, attendre le "R", enfoncer le bouton "cmd", attendre le "R", envoyer un "K" avec la clé, le keyer indique alors le mode sélectionné, par exemple KB pour le mode lambic B. On passe d'un mode à l'autre au moyen du contact "trait". Lorsque l'on entend de nouveau le "V", appuyez sur le bouton "cmd". On est alors dans le mode bug ; cela est confirmé par l'envoi d'un "R" par le keyer. Les points se font

Fig. 7.



connector heb ik er niet afgesloopt, die is onnodig maar blijft gewoon verder zitten.
De connector ICP in het schema is ook uitsluitend nodig om de controller te programmeren, tijdens het ontwikkelen van het programma en als je een geprogrammeerde chip hebt dus overbodig.

Sleutelen van de zender

Dat kun je op drie manieren doen. Met tussenschakeling van een reed relais of elektronisch. In dat laatste geval moet je weten wat de open spanning is op een seinsleutel als die niet gesloten is. Positief of negatief ten opzichte

alors automatiquement et les traits aussi longtemps que l'on maintient le contact "trait" fermé.

Tout n'a pas directement fonctionné comme souhaité. Pour le débogage, j'ai raccordé un afficheur LCD au port 2 au moyen d'un connecteur à 16 broches (**figure 6**). Les routines pour le dépistage des erreurs se trouvent à la rubrique "diagnostics" dans le programme enregistré.

Pour arriver à en savoir plus sur ces rebonds qui se produisent lors de changement de position du contact de la clé, j'ai écrit une routine d'interruption alternative qui permet d'établir un diagnostic.
Lors de la production d'une série de dits, le nombre de rebonds et la durée de rebond la plus longue sont mesurés chaque fois que le contact est établi ou rompu. Après 7 secondes, on obtient sur l'afficheur la durée maximum d'un rebond et le nombre maximum de rebonds par changement de contact.

Les clés ordinaires produisent un ou deux rebonds, mais si l'on fait du morse en mettant en contact deux fils l'un contre l'autre, on a un nombre élevé de rebonds. Ce 128 choisi pour la durée d'un rebond (16 ms) offre donc une large marge de sécurité. L'organigramme de la routine d'interruption alternative est donné à la **figure 7**.

Lorsque le programme est au point et fonctionne bien, l'afficheur LCD et la connexion au port P2 du processeur ne sont évidemment plus nécessaires. Le connecteur est toujours disponible, je ne l'ai pas mis au rebut.

Le connecteur ICP mentionné sur le schéma est uniquement nécessaire pour programmer le contrôleur et pour le développement du programme ; il est bien sûr superflu si vous disposez d'un CI programmé.

Manipulation de l'émetteur

Cela peut se faire de trois manières, par l'intermédiaire d'un relais reed ou électroniquement. Dans ce dernier cas, il est nécessaire de connaître la tension d'ouverture d'une clé morse lorsque celle-ci n'est pas fermée ainsi que l'intensité du courant lorsque la clé est fermée. Il faut aussi savoir si elle est positive ou négative par rapport à la masse.
La **figure 8** donne une possibilité de raccordement dans le cas ou la tension commandée est positive par rapport à la masse. Evidemment, le transistor

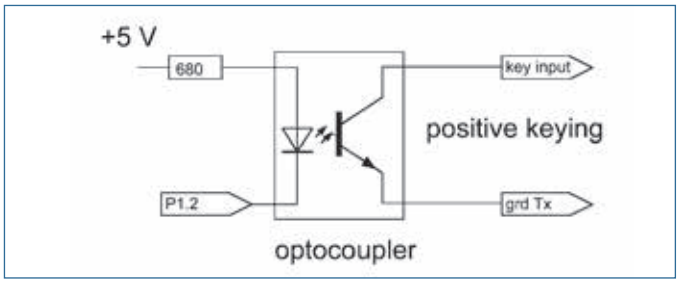
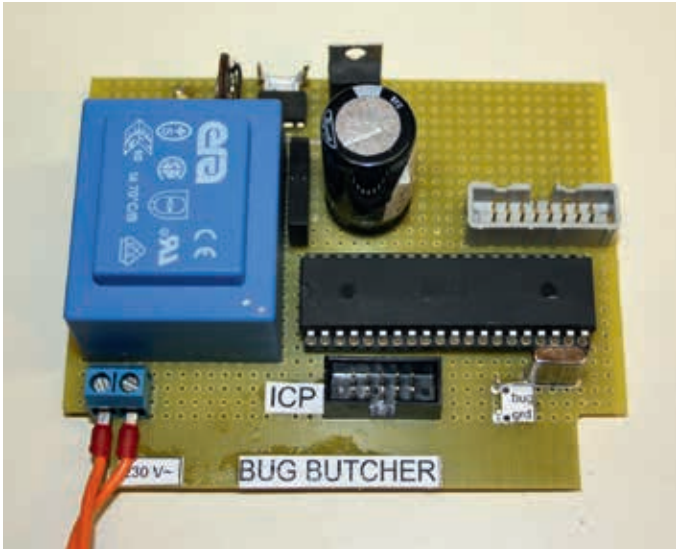
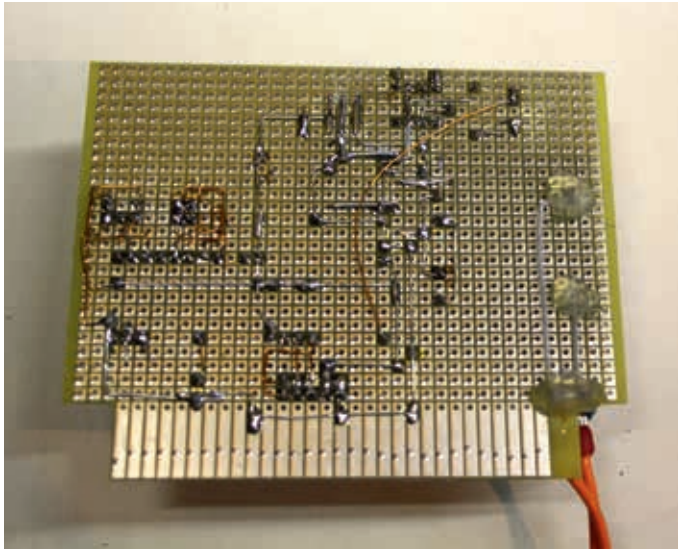


***Fig. 8. TX interface** / Interface avec le TX*





van massa, dat is belangrijk te weten, en vervolgens hoeveel stroom er loopt als de sleutel gesloten. De **figuur 8** geeft een mogelijke schakeling als je een positieve spanning naar aarde schakelt. Uiteraard moet de transistor in de optocoupler die spanning kunnen verdragen en de stroom bij een mark.

Bouw

De zaak is op gaatjesprint gezet en aan de achterzijde doorverbonden met draadjes 0,4 mm posijndraad of blank montagedraad. Als je een schakeling eenmalig maakt, is een dergelijke werkwijze naar mijn smaak het beste. Printen zijn voor massafabricage, die zijn niet geschikt, want je kunt niets of nauwelijks iets wijzigen, dus experimenteren is dan taboe.

Voor nabouwers kan ik een microcontroller programmeren, dat kost je inclusief porto 15 euro. Neem indien je dat wilt contact op via e-mail met mijncall@amsat.org, waarbij je mijncall uiteraard dient te vervangen door

PA0WV

de l'optocoupleur doit pouvoir supporter cette tension ainsi que le courant lors d'un mark.

Construction

L'ensemble est monté sur une plaquette à trous ; les liaisons sont faites à l'arrière avec du fil de câblage étamé ou avec des fils à souder de 0,4 mm. Pour un montage unique, cette méthode me semble être la meilleure. Les circuits imprimés sont intéressants pour des réalisations en grande quantité. Ils ne sont pas intéressants ici car il est difficile, si pas impossible, de faire des modifications et donc d'expérimenter.

Pour ceux qui souhaitent se lancer dans la construction de l'appareil, je peux programmer un microcontrôleur pour le prix de 15 euros, port inclus. Prendre alors contact avec moi par mail à l'adresse moncall@amsat.org, dans laquelle vous remplacez évidemment moncall par :

PA0WV

VHF

VHF Manager	ON6TI - Stefan	on6ti@uba.be
VHF Manager (Assistant)	ON4AVJ - Jacques	on4avj@uba.be
VHF Microwaves Manager	ON7BPS - Peter	on7bps@uba.be
VHF 6m Band Manager	ON4IQ - Johan	on4iq@uba.be
VHF Contest Manager	ON4AVJ - Jacques	on4avj@uba.be
VHF Technical Manager	ON4PC - Filip	on4pc@uba.be
VHF Digital Modes Manager	ON4PN - Patrick	on4pn@uba.be
VHF EME Manager	ON4KNG - Peter	on4kng@uba.be
VHF Satellite Manager	ON4HF - Eric	on4hf@uba.be

VHF Propagatie in theorie en praktijk (slot)

door ON4AVJ Jacques

In dit laatste deel gaan we een aantal bijzondere propagatie media bespreken.

Deel 5: Andere propagatiemedia

Sporadic E

Een van de meest spectaculaire propagatie modes is Sporadic E of afgekort Es. Een bijna dode band komt plots tot leven en je kan stations werken die zeer veraf zijn. Afstanden tot 8000 Km zijn mogelijk op 50 MHz en tot bijna 4000 km op 144 MHz. Sporadic E kan voorkomen tot ongeveer 200 MHz. Een paar zeldzame keren werd in de US Es gemeld in de 220 MHz Band (niet toegestaan in Europa). Deze signalen verdwijnen plots op dezelfde wijze zoals ze verschenen.
Het is ook een zeer lokaal verschijnsel. Je kan met jouw station andere tegenstations werken uit meestal dezelfde locator square of een paar naburige squares met 59+ terwijl een ander station een paar 10-tal km van jou verwijderd niets hoort. Het kan ook gebeuren een station een paar 100 km van jou verwijderd totaal andere stations werkt in totaal verschillende locatorsquares.

Wat is het?
In de E-laag zijn er naast atmosferische gassen ook soms hoge concentraties van metaalatomen aanwezig. Deze atomen zijn afkomstig van het opbranden van meteorieten in de atmosfeer. Deze metaalatomen hebben veel minder energie nodig om geïoniseerd te worden. De intense zonnestralen (UV) in de zomer zijn dus in staat deze metaalatomen te ioniseren.
De concentratie van zulke metaal atomen worden ook wel eens "Es Wolken" genoemd. Deze wolken kunnen voorkomen op verschillende hoogtes

Propagation VHF en théorie et en pratique (fin)

par ON4AVJ Jacques – traduit par ON5FM Guy

Dans cette dernière partie, nous allons parler d'un certain nombre de modes de propagations spéciaux.

5ème Partie : Autres modes de propagation

Sporadic E

Un des modes les plus spectaculaires est le Sporadic E ou, en raccourci, l'Es. Une bande presque morte peut brusquement revenir à la vie et vous pouvez alors contacter des stations très éloignées. Des distances jusque 8000 Km sont possibles sur 50 MHz et jusque presque 4000 Km sur 144 MHz. Le Sporadic E peut agir jusque 200 MHz. A quelques rares occasions, des ES ont été signalées dans la bande des 220 MHz (non attribuée en Europe). Les signaux disparaissent aussi subitement qu'ils sont apparus.
C'est aussi un phénomène local. Vous pouvez contacter des stations de votre carré locator ou des carrés voisins avec des rapports de 59+ tandis qu'une autre station située à quelques dizaines de km de vous ne peut rien entendre. Il peut aussi se produire qu'une station à quelques centaines de km de vous contacte des stations complètement différentes et dans des carrés locator totalement différents.

De quoi s'agit-il ?
Dans la couche E, il y a parfois, dans les gaz atmosphériques voisins, une relativement haute concentration d'atomes métalliques qui est présente. Ces atomes sont issus de la combustion des météorites dans l'atmosphère. Ces atomes de métaux nécessitent beaucoup moins d'énergie pour être ionisés. Le rayonnement solaire intense (en UV) en été est donc en mesure d'ioniser ces atomes métalliques. La concentration de tels atomes métalliques est aussi parfois appelée "nuages Es". Ces nuages peuvent apparaître