DSC analyzer, version SX64

For at være i stand til at teste på <u>DSC</u> radioudstyr^{*1} blev programmet **DSC-analyzer** udviklet i 80erne.

Først kodet til en Commodore SX64 (Fig.1), der som C64 bl.a. udmærker sig ved at have en yderst versatil CIA-chip^{*2} der f.eks. kan sættes til at opsamle datastrømme, timet af en præcis ekstern reference.

Programmet er dels kodet i Assembler^{*3} for timing-kritisk opsamling og afsendelse af DSC-beskeder på bitniveau, samt i højniveausproget Comal80 som brugerinterface (Fig.2), og præsentation af data (Fig.3).

Koden blev efterhånden så omfattende, at den måtte deles op i moduler, der loades inden programstart (Fig.4).



Fig.1

^{*1} Digital selective calling or DSC is a standard for transmitting pre-defined digital messages via the maritime radio systems. It is a core part of the Global Maritime Distress Safety System (GMDSS). <u>Kilde</u>

^{*2} Complex Interface Adapter MOS6526

*3 CPU MOS6510 Kilde

^{*4} FEC (Forward Error Correction) The sender encodes the message in a redundant way by using an error-correcting code (ECC). <u>Kilde</u>

^{*5} ECC (Error Correction Character) er en modulo2 sum af de modtagne primære databit (DX) samt de redundante databit (RX). <u>Kilde</u> <u>Se evt. også Fig.5</u>

*6 UniComal er en object orienteret version af Comal80





Fig.2 Brugerinterface

	126 041	126		127	izi i			127	127
ECC:	dx :	41 P	< : 41				%10		1001
D 85 R111	85 110	125	125 108	125 107	125 106	125	125 104	116	116
R118	89 89	21 21	91 91	80 80	80 80	109 109	111	126	126
R126	126	126 126	126 126	127	壮	127	127		

Fig.3 Modtaget DSC-besked. Øverst de modtagne bit (7 databit + 3 paritetsbit) på decimal form, og nederst korrigeret for FEC^{*4}-forskydning samt beregnet ECC^{*5}.



Fig.4 Programmet kunne efterhånden ikke være editorhukommelsen, og måtte deles op i moduler.

```
FUNC ecc#(dxrx$)
  n:=phas_start+24 // 1.format spc
  IF dxrx$="dx" THEN
    ec#:=call#(n)
    n:+8
  ELSE
    n:+10
    ec#:=call#(n)
    n:+8
  ENDIF
  ec#:=ec# BITXOR call#(n)
  REPEAT
    n:+4
    ec#:=ec# BITXOR call#(n)
    eos_test#:=eos#(n,dxrx$)
  UNTIL eos_test# OR eos_test#=-1
  RETURN ec#
ENDFUNC ecc#
```

Fig.5 Eksempel på Comal80-kode. Her funktionen til beregning af <u>ECC</u>^{*5} modulo2 sum i **DSC-analyzer**.

	8292	20	07	A.a	001001	ldv	will of				
	asas	00	00	00	an an ai an an	1 U A	40				
	0007	20	00		0000	CHX	11 C)				
	0697	00	00			one 	rii.				
	6699	a9	Эa	1000 - 111 V		ida	#farve3				
	853P	50	q5	ff		jsr	chrout				
	829e	4c	b9	82		jmp	skriv				
	82ai	eO	09		ni	срх	#9				
	82a3	dO	08			bne	ti				
	82a5	a9	9a			1da	#farve3				
	82a7	20	de	ff		ism	chrout				
	APaa	40	69	82		imm	eboiv				
	apad	-0	0.5	See Less	4-3	June mmv	SKI IV				
	00.4	20	00		6 L	CPX	TT L C				
	ocal oct	-0	00			Drie	SKriv				
	8201	89 00	9a			Ida	#tarve3				
	8603	20	Cie	T T		Jsr	chrout				
	8508	4·C	p.a	82		jmp	skriv				
	8269	a5	a6		skriv	lda	\$a6				
	85PP	50	d2	ff		jsr	chrout	;jsr	udsk	riv	
	82be	ae	07	8a		ldx	nul et	1.4			
	82c1	eO	Oa			CDX	#10	:10 h	nit?		
	82c3	fO	06			hen	nem bytes	920 8	<i></i>		
	the first the thr					tar tar	gem_bytes				
	00A5	20	20	07		i	de á anos er er da				
	0.00	<u> </u>	0C	00		Jer.	cimeout				
	occo	~+ C.	40	oc.		Jmp	timstart	;one	Dit	more	
	.85cp	96	06	8a	gem_bytes	ldx	husk_x				
	82ce	18				clc					
	82cf	6e	00	8a		ror	rul_1_7	rul	endn	u et	0
	82d2	a5	c5		chktast	lda	\$c5				
	82d4	c9	40			emp	#64				
	82d6	fO	07			hen	er ryslut				
	ADAA	 . 9	00			1	40				
	ood-	0E				T CI CI	m0 *				
	00de	<u> </u>	24	00		blet	*C6				
	camur.	-+C	11.	oc		Jmp	tli_comai				
			~ .	-							
	8eor	ao	01	8a	er_rxs1ut	lda	rul_8_a				
	8565	25	07			and	%00000111				
	82e4	cd	18	8a		cmp	rxslutbits				
	82e7	dO	13			bne	not_retur				
P	82e9	ac	00	8a		ldy	rul 1 7				
	82ec	cc	18	8a		CDY	rxslutbits				
	82ef	dO	Оb			bne	not retur				
	82f1	a9	FF		til comal	lda	#%11111111				
	8263	94	00	90	ar ah ah _{ana} ∖as \s/111€35 d.		wydstaatt				
	00 <i>4C</i>	20	vv	20			i Aududrein, X				
	0000	80	~~	00		INX					
	ocr/	90	00	90		sta	rxdataram, x				
	ocra	38				cli					
	SETD	60				rts					
	82fc	ad	00	8a	not_retur	lda	ru1_1_7				
	82ff	9d	00	90		sta	rxdataram, x				
	8302	e8				inx					
	8303	8e	06	8a		stx	husk x				
	8306	ad	01	8a		lda	rul A a				
	8309	Sci	00	90		at a	pydatapam y				
	8300	aQ	~~			ive	r voereeu.emil x				
	Q 204	00	06	Q -		TLIX	han me				
	0000	OE	00	Oct		BCX	nusk_x				

Fig.6 Som eksempel på assemblerkoden, ses her side 6 af i alt 9 A4-sider.

DSC analyzer, version PC

I starten af 90'erne blev programmet migreret til PC (Fig.7 til 9), nu med UniComal^{*6} som højniveau-sprog, og timingkritisk data-opsamling og afsendelse skrevet i 8086-maskinkode. I mangel af en dedikeret I/O-port, benyttes i stedet signalbenene på PC'ernes RS232-stik.

På linje med alt andet testudstyr, var programmet underlagt laboratoriets akkrediteringskrav, og afslørede gennem tiden utallige afvigelser fra gældende standarder på det udstyr det testede, hvorfor firmaerne viste interesse for at købe det, hvilket dog var mod laboratoriets formål, nemlig at sælge målinger, og ikke SW.

Da laboratoriet var med i internationalt standardiseringsarbejde, blev der på et tidspunkt behov for at afdække sammenhængen mellem BIT-error (BER) og MESSAGE-error (MER) på forskellige beskedtyper, og forskellige DSC-modtagere, visende hvor godt fabrikanterne formåede at udnytte dataformatets indbyggede fejlkorrektioner (FEC^{*4}, ECC^{*5} samt paritetsbit).

Der blev derfor udviklet en udgave at programmet, hvor indsættelse af et stigende antal tilfældige bitfejl i afsendte beskeder simulerede dårlig radiodækning, og for at vide hvordan det testede udstyr (EUT) tolkede de afsendte beskeder, skulle programmet også simulere EUT-printer, dvs. modtage og tolke det EUT'en udskrev, hvilket der kom en 75-siders tyk rapport ud af, hvoraf et par sider kan ses fra <u>side 9</u>.

Alt i alt en udfordrende men yderst interessant programmeringsopgave.

🖧 DSC272	
Auto 💽 🛅 🛍 🔂 🗃 🚰 🔺	
DSC analyzer=-	∎┤┤┤┤┤┤┤┤┤
f1 Receive F2 TX (Space)	
f3 Load call f4 Save call	
f5 Display f6 Display Bin	
Compose F8 Send many	
f9 Send mark f10 Send space	
File Data files file SetParameter	┛┤┤┤┤┤┤┤┤┤
ChangeModem END Load/Dos/END	
	┛┤┤┤┤┤┤┤┤┤
ver 2.72	
AltF1 NumOfDis - CtrlF1 DisTimeDis	
AltF2 TXthenRX - CtrlF2 Repeat TX	
ActiveCall: No11 All For dx1 rx2 Par	
CallMem: 1 <press tab=""></press>	
MF/HF modem	

F1, **Receive:** Programmet går i RX mode og venter på data tilført RS232 CTS (ben 8 & 5, hvis DB-9).

F2, TX: Programmet sender de data der er i hukommelsen ud på RS232 DTR (ben 4 & 5, hvis DB-9.

F3, Load call: Henter data fra fil.

F4, Save call: Gemmer data til fil.

F5, **Display:** Viser de data der er hentet via CTS eller fra fil. Data vises dekodet i klar tekst, og som de er modtaget enten i decimalt eller hexadecimalt talsystem. Det er muligt at editere data. Se Fig.8, og beskrivelsen side 7.

F6, Display Bin: Som 'F5' men data vises binært.

F7, Compose: Komponer en besked.

F8, Send many: Sender et valgt antal kald fra fil, med valgt pausetid imellem.

F9, Send mark: Sætter COM1 DTR høj, og starter en tone på 1615Hz (MF/HF) / 1300Hz (VHF) i PC'ens højttaler.

F10, Send space: Sætter COM1 DTR lav, og starter en tone på 1785Hz (MF/HF) / 2100Hz (VHF) i PC'ens højttaler.

F11, Data files: Kopierer og opretter data filer.

F12, SetParameter: Her defineres vigtige programparametre, bl.a. stien til data filerne.

M, ChangeModem: Vælg mellem MF/HF og VHF-mode.

END, **Load/Dos/END**: Tast to gange på 'End' tasten for at afslutte programmet.

<Alt>F1, NumOfDis: Tæller antallet af 'distress' gentagelser. (Krav fra standard).

<Ctrl>F1, DisTimeDis: Måler tiden der går mellem hvert distress kald. (Krav fra standard).

<Alt>F2, TxthenRX: Sender data og går derefter i receive mode. (Nyttigt for test af kaldtyper der udløser reply).

<Ctrl>F2, Repeat TX: Sender samme kald et antal gange.

F3, Load call

🔀 DSC272			_ 🗆 🗵						
	MfHf Normal	MfHf Error	┓┓┥						
	<pre>0 No 0 Individual Routine 1 No 1 AllShip Urg Medical 2 No 2 GeoArea Rut RQ 3 No 3 Polling Indiv RQ 4 No 4 Shippos Indiv RQ 5 No 5 SemiAuto Even RQ 6 No 6 SemiAuto Odd RQ 7 No 7 Distress Undesignated 8 No 8 Distress Acknowledge 9 No 9 DistRelay AllShp RQ 10 No10 DistRelay AllShp RQ 11 No11 DistRelay AllShp RQ 12 No12 All Ship REC586 13 No13 Individual REC586 14 No14 Indiv ShipBu Pay RQ 15 No15 Idv ShipBu RXchn RQ 16 17 18 * 20 answer to auto 21 dsc on work tx semi answer to auto 2 23 24 25 26 27 27 28 29 30 31 32 33 34 35 No35 Distress Fire Exp <return> <pgun> <pgdn> <crsrup <home> <end> <esc> <d>ot</d></esc></end></home></crsrup </pgdn></pgun></return></pre>	<pre>0 No 0 Pha 2dx & 1rx ok 1 No 1 Dis For rx1 rx2 par 2 No 2 Aut ECC dx parity 3 No 3 All For rx1 rx2 par 4 No 4 Ind For rx1 rx2 par 5 No 5 Dis ECC dx Parity 6 No 6 Ind ECC dx Parity 7 <> No 7 All ECC dx Parity 8 No 8 Pha 1dx & 2rx ok 9 No 9 Dis For dx1 rx2 Par 10 No10 Aut ECC rx Parity 11 No11 All For dx1 rx2 Par 12 No13 Dis ECC rx Parity 13 No13 Dis ECC rx Parity 14 No14 Ind ECC rx Parity 15 No15 All ECC rx Parity 16 No16 Pha 0dx & 3rx ok 17 No17 Dis For dx2 rx2 Par 18 No18 Aut ECC dx & rx Par 19 No19 All For dx2 rx2 Par 20 No20 Ind For dx2 rx2 Par 21 No21 Dis ECC dx & rx Par 22 No22 Ind ECC dx & rx Par 23 No23 All ECC dx & rx Par 24 No24 Dis For rx1 dx2 Par 25 No25 Aut ECC dx Num&Par 26 No26 All For rx1 dx2 Par 27 No27 Ind For dx1 dx2 Par 28 No28 Dis ECC dx Num&Par 30 No30 All ECC dx Num&Par 31 No31 Dis For dx1 dx2 Par 32 No33 Ind For dx1 dx2 Par 33 No33 Ind For dx1 dx2 Par 34 No34 Dis For dx1 dx2 Par 35 No35 All For dx1 rx1 Par 35 No35 All For dx1 rx1 Par 36 No35 All For dx1 rx1 Par 37 No37 Ind For dx1 rx1 Par 38 No38 All For dx1 rx1 Par 39 No39 All ECC dx Num&Par 30 No30 All ECC dx Num&Par 31 No31 Ind For dx1 dx2 Par 32 No35 All For dx1 dx2 Par 33 No35 All For dx1 dx2 Par 34 No34 Dis For dx1 dx2 Par 35 No35 All For dx1 rx1 Par 36 No35 All For dx1 rx1 Par 37 No35 All For dx1 rx1 Par 38 No35 All For dx1 rx1 Par 39 No39 All For dx1 rx1 Par 30 No39 All For dx1 rx1 Par 30 No39 All For dx1 rx1 Par 31 No39 All For dx1 rx1 Par 32 No35 All For dx1 rx1 Par 33 No35 All For dx1 rx1 Par 34 No35 All For dx1 rx1 Par 35 No35 All For dx1 rx1 Par 36 No35 All For dx1 rx1 Par 37 No35 All For dx1 rx1 Par 38 No35 All For dx1 rx1 Par 39 No35 All For dx1 rx1 Par 30 No35 All For dx1 rx1 Par 30 No35 All For dx1 rx1 Par 31 No35 All For dx1 rx1 Par 32 No35 All For dx1 rx1 Par 33 No35 All For dx1 rx1 Par 34 No35 All For dx1 rx1 Par 35 No35 All For dx1 rx1 Par 36 No35 All For dx1 rx1 Par 37 No35 All For dx1 rx1 Par 38 No35 All For dx1 rx1 Par 39 No35 All For dx1 rx1 Par 30 No35 All For dx1 rx1 Par 30 No35 All For dx1 rx1 Par 30 No35</pre>							

Fig.8. Eksempel på menuvalg 'F3, Load call' i MF/HF mode.

Venstre kolonne 'MF/HF Normal' ('VHF Normal') indeholder fejlfri DSC-kald.

Højre kolonne 'MF/HF Error' ('VHF Error') indeholder DSC-kald med bitfejl tilføjet mere eller mindre kritiske steder i bitstrømmen, med henblik på at teste en DSC-dekoders reaktion på den fejlbehæftet bitstrøm. (Krav fra standard).

Markøren '<>' peger på kald no. 7, der er et 'All ships call' med påført paritetsfejl i ECC karakterens DX del. Se Fig.9.

Der skiftes mellem kolonnerne med højre/venstre piletast.

PS! Dette 'pseudofilsystem' er designet for at kunne bruge lange filnavne (der ikke var opfundet da programmet blev lavet), og arbejder med data fra følgende fire fysiske filer: MFNORM.RAN, MFERROR.RAN, VHFNORM.RAN, samt VHFERROR.RAN. Placeringen af disse filer skal fremgå af 'File path' under '**F12, SetParameter**'. 'File path' er default det samme katalog som programmet ligger i / starter fra.

F5, Display



Fig.9. Eksempel på menuvalg 'F5, Display'.

Venstre vindue: Her vises data i klar tekst, 'dekodet' i overensstemmelse med standarden.

Højre vindue, øverste del: Her vises data i den rækkefølge de er modtaget, i decimalt eller hexadecimalt talsystem. Hvert tal repræsenterer 10 bit (7 databit + 3 paritetsbit). Er der uoverensstemmelse mellem databit og paritetsbit (paritetsfejl), markeres dette som highlight af det berørte tal (her **041**).

%1011111001 viser det markerede tal *1 på binær form *2. Ved at trykke <Enter> kan det binære tal editeres. Ved fornyet tryk på <Enter>, opdateres det markerede tal, samt dekodningen i venstre vindue.

Indtastes der et trecifret tal, f.eks. '012' (uden brug af <Enter>), opdateres det markerede tal (og dekodningen i venstre vindue) direkte. (Tocifret tal, hvis i 'hex' mode).

*1 Markeringen vises som en cyan-farvning af midterste tal (her 2-tallet i tredjeførste 125).

*2 De syv databit (**1011111**) er vist bagfra (mindst betydende bit først) da de, af grunde jeg har glemt, fysisk sendes på denne måde 'i luften'. De tre paritetsbit (*001*) angiver antallet af '0'-er i databit-delen.

ECC: dx:41 rx:41 viser resultatet af en modulo2 addition af alle data, på nær selve ECC-karakteren (som beskrevet i standarden).

Højre vindue, nederste del: Her vises data, parret DX/RX.

Er der uoverensstemmelse mellem DX og RX, markeres dette som en farveændring af RX-delen.

NB! Data vil blive sendt (F2, TX) og gemt (F4, Save call) som de vises under dette menupunkt.

Test Report of correlation between SER og BER

By Lars Hansen

Ref. No. Client Contact person

7/1-94

Date and signature



Call type: Distress

No of		SER-EUT1	SER-EUT2
BitError	BER	SER	SER
0	0,00	0,00	0,00
4	0,74	0,49	0,41
5	0,93	0,81	0,64
6	1,11	1,34	0,85
7	1,30	1,78	1,11
8	1,48	2,18	1,76
9	1,67	2,89	2,07
10	1,85	3,27	2,73
11	2,04	4,02	3,15

EUT1 = EUT2 =

Example of a call supplied with errors, indicating the bit error position

FormatSp Individual stations 120 Address. 123456789 Category Routine 100 StatioID 002191000 1.teleCo No information 126 2.teleCo No information 126 Frequenc RX:--,--- TX:--,---EndOfSeq No ackn. neded (EOS) 127

1010101010101010101010111110011111011001
101111100101110110101011111001101101101
101111100100110110111011111001110101101
101111100101010110110001111011100101101
0001111011000101110000110001010001111011
0100010101000111101100011101000011000101
0111001011010001010101010101010001110100
0010011100011100101100000001110101101011
1010100100001001110011011010100000000111
0000000111101010010000000001111101101010
01111110010000001110111111001000000111
0111111001011111100101111110010111111001
0111111001011111100101111110010111111001
0111111001011111100101111110010111111001
1111111000011111100111110100100111111001
1111111000111111100011111110001111010010

Call: Init call

1993-11-03 11:41:23 EUT1 40*16=640 bits with no errors. CallNo, ReceivedChars, NoOfErr, Bit position with error.

Init	call	640	5	21	0,78			
	l	21	0	472	79	191	119	136
	2	21	0	475	381	243	218	320
	3	21	0	182	306	393	399	544
	4	21	0	210	613	348	553	3
	5	21	0	538	598	416	58	158
	6	21	0	245	522	566	239	381
	./	21	0	618	376	209	445	468
	8	21	0	305	361	277	589	622
	9	21	0	231	623	357	431	92
	10	21	0	13	124	137	480	44 / E 4 7
	11	21	0	101	431	242	170	120
	12	21	0	126	300	243	1/0	281
	13	21	0	130	105	510	520	204
	14	21	0	376	280	375	117	424
	16	21	0	200	557	483	160	121
	17	21	1	572	411	126	366	208
	18	21	0	228	224	438	315	221
	19	21	1	575	149	588	496	444
	20	21	0	633	325	247	107	501
	21	21	0	558	587	328	589	611
	22	21	1	471	541	325	241	597
	23	21	1	618	447	161	215	603
	24	21	0	11	54	327	442	156
	25	21	0	384	548	610	8	243
	26	21	1	161	263	215	515	346
	27	21	1	309	170	50	490	353
	28	21	0	277	616	172	378	460
	29	21	0	74	271	499	282	633
	30	21	0	447	125	143	488	80
	31	21	0	224	480	387	355	183
	32	21	0	105	240	507	535	277
	33	21	0	362	434	479	353	551
	34	21	0	600	50	370	135	116
	35	21	0	333	543	13	340	203
	36	21	0	98	4	462	132	483
	37	21	0	200	359	67	640	586
	30	21	0	390	221	100	10U 610	220
	39	21	1	579	576	240	480	121
	40	21	0	86	482	76	454	430
	42	21	0	459	336	359	20	517
	43	21	0	236	51	604	528	621
	44	21	0	477	65	439	355	172
	45	21	0	358	466	558	535	265
	46	21	0	615	20	531	353	539
	47	21	0	213	275	421	134	104
	48	21	0	585	129	65	340	191
	49	21	0	363	484	309	208	295
	50	21	0	575	177	321	363	552
	51	21	0	533	632	280	317	555
	52	21	0	310	348	524	185	18
	53	21	0	639	394	494	203	390
	54	21	0	596	209	452	157	392
	55	21	0	521	471	533	639	501
	56	21	0	94	164	544	155	119

Individual call.

CallNo, ReceivedChars, NoOfErr, Bit position with error.

317	21	0	379	542	512	550	247
517	21	0	150	057	770	410	250
348	21	0	120	257	110	418	350
349	21	0	529	111	400	624	437
212	01	0	200	ACC	1	101	511
350	21	0	300	400	4	491	741
351	21	0	352	367	385	111	384
252	21	0	EOO	220	572	163	274
352	21	0	592	520	515	405	2/1
353	21	0	157	389	585	133	587
DEA	21	0	115	581	558	591	221
354	21	0	410	504	550	551	222
355	21	0	147	437	201	157	308
356	21	0	285	353	514	197	209
550	21	0	205	500	241	207	477
357	21	0	5	592	362	321	4/3
358	21	0	211	14	374	637	145
250	01	0	100	170	E 2 2	11	52
359	ZI	0	490	4/0	525		52
360	21	0	68	169	534	167	310
361	21	0	126	324	71	502	616
301	21	0	120	221	216	270	70
362	21	0	544	39	316	370	19
363	21	0	276	533	599	575	167
200	01	0	E 2 4	07	570	202	440
364	ZI	0	554	0 /	572	555	110
365	21	0	266	581	215	599	528
366	21	0	2	133	541	336	384
500	21	0	1.00	100	227	421	60
367	21	0	169	597	331	431	69
368	21	0	542	451	620	637	156
200	21	1	107	512	633	306	468
309		Ξ.	107	JIZ	055	500	400
370	21	0	65	349	74	176	221
371	21	0	232	174	510	271	546
571	21	0	102	104	0.0		1 5 1
372	21	0	483	104	98	2	TOT
373	21	0	170	89	166	146	305
271	21	0	238	213	383	500	326
3/4	ZI	0	230	215	505	500	520
375	21	0	546	320	271	482	628
376	21	0	233	305	338	626	142
570	21	0	200	120		240	1 ()
311	21	0	301	430	220	540	102
378	21	0	91	460	498	157	307
270	21	0	230	315	212	215	563
519	21	0	255	545	212	210	202
380	21	0	566	330	280	360	11
381	21	0	97	500	372	385	87
202	01	0	101	105	110	E20	212
382	21	0	424	400	440	550	242
383	21	0	157	339	83	95	329
381	21	1	504	263	233	276	552
201	21	+ 0	001	240	100	255	171
385	21	0	202	240	464	355	1/1
386	21	0	70	325	246	586	224
207	21	0	112	179	529	151	312
301	21	0	442	119	525	101	512
388	21	0	130	164	597	296	466
389	21	0	342	497	608	452	83
202	01	0	570	140	4 5 0	215	E 7 4
390	21	0	570	142	400	215	574
391	21	0	258	128	526	360	88
200	21	1	470	460	537	515	345
352		1	110	100	100	10	515
393	21	1	348	537	482	40	503
394	21	0	406	73	141	292	560
205	01	0	622	350	621	55	477
332	21	U	033	555	051	55	
396	21	0	321	344	58	200	565
397	21	0	533	37	70	355	182
200	21	0	107	100	100	500	225
398	21	0	401	122	492	200	233
399	21	0	133	616	135	152	322
400	21	0	461	601	203	296	477
TUU	21	U	101	001	200		