NASA CR-152580

TDRSS TELECOMMUNICATION SYSTEM PN CODE ANALYSIS

FINAL REPORT ADDENDUM

(NASA-CR-152580) TDRSS TELECOMMUNICATION SYSTEM PN CODE ANALYSIS Final Report (Gold (Robert) Associates, Los Angeles, Calif.) 90 p HC A05/MF A01 CSCL 17B

N77-30318

Unclas
G3/32 46127

Contract NAS 5-22546

Prepared for

National Aeronautics and Space Administration
Goddard Space Flight Center
Greenbelt, Maryland 20771

SEP 1977
RECEIVED
NASA STI FACILITY
INPUT BRANCH

Prepared by

Robert Gold
Robert Gold Associates
435 South La Cienega Boulevard
Los Angeles, California 90048

April 1, 1977

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1.0 INTRODUCTION

This report contains the pseudonoise (PN) code library for the Tracking and Data Relay Satellite System (TDRSS) Services. This library has been defined as a result of work carried out by Robert Gold Associates for NASA Goddard Space Flight Center under Contract NAS 5-22546.

The code library contained herein is chosen to minimize user transponder hardware requirements and optimize system performance. Special precautions were taken to insure sufficient code phase separation where required, to minimize cross-correlation sidelobes, and to avoid the generation of spurious code components which would interfere with system performance.

2.0 PN CODE LIBRARY FOR TDRSS

The PN code library for TDRSS services is listed in Table 1. Eighty-five code assignments are identified, with each code assignment consisting of PN codes for all forward and return link channels and for all modes of operation. Each user spacecraft will receive a code assignment number which uniquely defines (via Table 1) a set of dedicated PN codes. In what follows, the individual code libraries are described and the use of Table 1 is illustrated.

When code generator feedback taps are specified in octal notation in Table 1, the translation to actual feedback taps may be carried out as illustrated in the following example.

Octal notation:	1 0 2 2 0 0 5
Binary:	001 000 010 010 000 000 101
Polynomial:	$x^{18} + x^{13} + x^{10} + x^2 + 1$

The stage of the feedback shift register code generator which receives the output of the modulo-two adder corresponds to the x term of the polynomial; the next stage corresponds to the x^2 term of the polynomial, etc.

The jth stage of the shift register has a feedback tap if and only if the coefficient of the x^j term of the polynomial is not zero.

Column 1 of Table 1 designates the user number to which is assigned all the codes of that particular row.

Column 2 of Table 1 defines the user-unique initial conditions in binary notation of stages 3 through 9 of the A register generating the command channel PN code. The initial conditions for stages 1, 2 and 10 of the A register are always zero. (The stage that receives the feedback from the modulo-2 adder is designated as the first stage.) The corresponding initial conditions for the B register are always 1 0 0 1 0 0 1 0 0 0. The Gold codes defined by these initial conditions will be balanced in the sense that the number of ones in the code sequence exceeds the number of zeros by one. As an example of the use of this column, the initial conditions of the command channel PN code generator for user code assignment 12 is illustrated in Figure 1.

Column 3 of Table 1 lists the octal representation of the feedback taps for the range channel PN code generator. As an example, the feedback taps for the code generator for user code assignment 12 are listed as 1023103 (octal) ≡ 001 000 010 011 001 000 011 (binary) ≡ $x^{18}+x^{13}+x^{10}+x^9+x^6+x+1$ (polynomial). The configuration of the PN code generator for user code assignment 12 is illustrated in Figure 2.

Column 4 is an octal definition of the feedback taps for the Mode 1 return link PN code generator. The configuration of the PN code generator for user code assignment 12 is illustrated in Figure 3. The modulo-2 sum of the I channel code and the tap from the ninth stage of the shift register is used as the Q channel code. The codes chosen have the following properties:

(a) The phase difference between the I and Q channel codes will be in excess of 20,000 chips. The exact phase difference in chips between these two codes for each user is listed in Column 5.

(b) The spurious codes generated due to filtering and limiting of the I and Q channel codes will be phase shifts of each of these codes.

The relative phase between the spurious codes and the I and Q channel codes will be in excess of 5000 chips.

Column 6 of Table 1 contains the octal representation of the initial conditions for the eleven stages of the A register of the shift register configuration generating the Mode 2 return link I channel code. The initial conditions of the B register must be 001 (octal) ≡ 0000000001 (binary) as shown in Figure 4.

Column 7 of Table 1 contains the octal representation of the initial conditions for the eleven stages of the C register of the shift register configuration generating the Mode 2 return link Q channel code.

When the code pairs determined by the initial conditions of columns 6 and 7 are used for the I and Q channels of the Mode 2 return link, the spurious Gold codes generated on these channels due to filtering and hard-limiting will not be members of the code library of Table 1.

All Mode 2 return link codes are balanced.

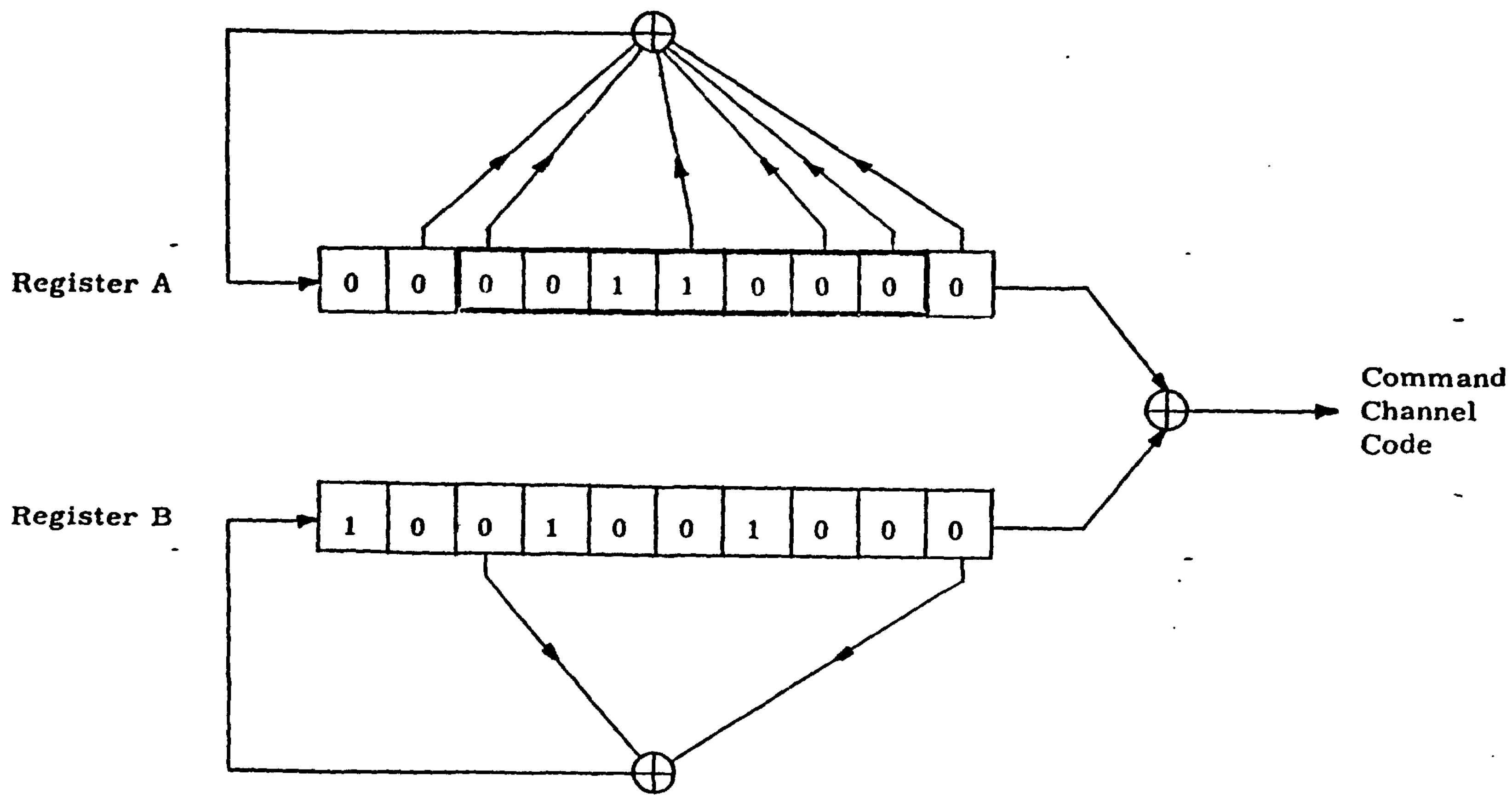

Figure 1. Command Channel Code Generator for User Code Assignment #12

Octal $\equiv$ 1 0 2 3 1 0 3

Binary $\equiv$ 001 000 010 011 001 000 011

Polynomial $\equiv x^{18} + x^{13} + x^{10} + x^9 + x^6 + x + 1$

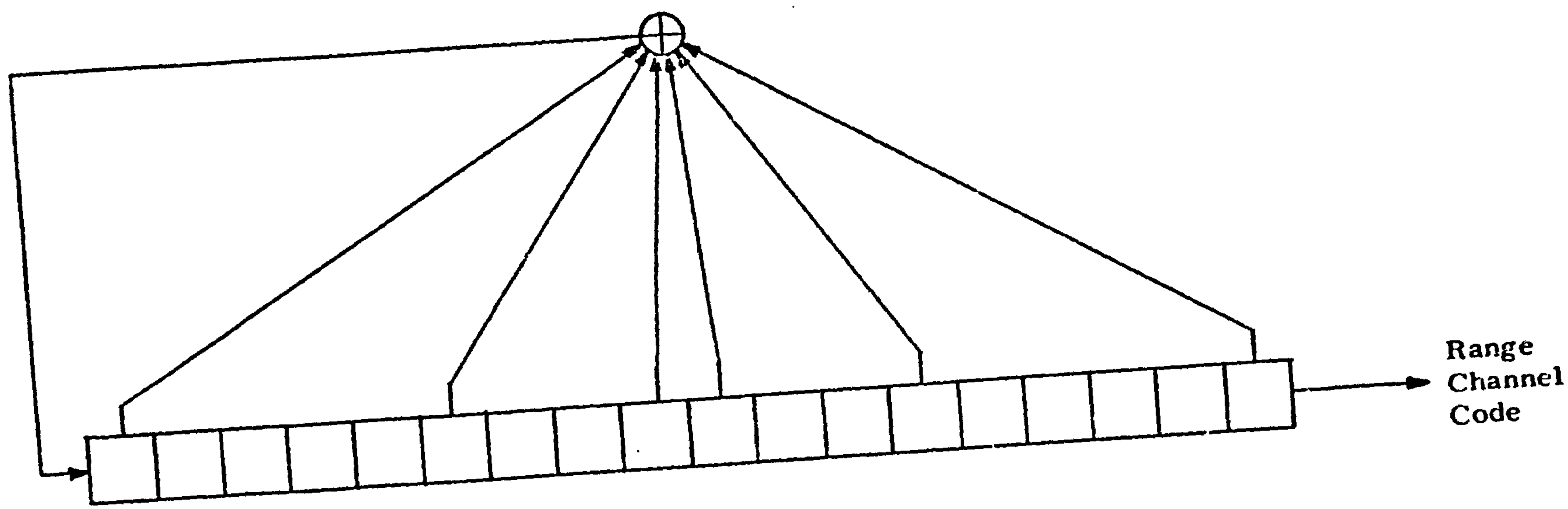

Figure 2. Range Channel Code Generator for User Code Assignment #12

Octal ≡ 1 0 1 1 3 3 3

Binary ≡ 001 000 001 001 011 011 011

Polynomial ≡ $x^{18} + x^{12} + x^{9} + x^{7} + x^{6} + x^{4} + x^{3} + x + 1$

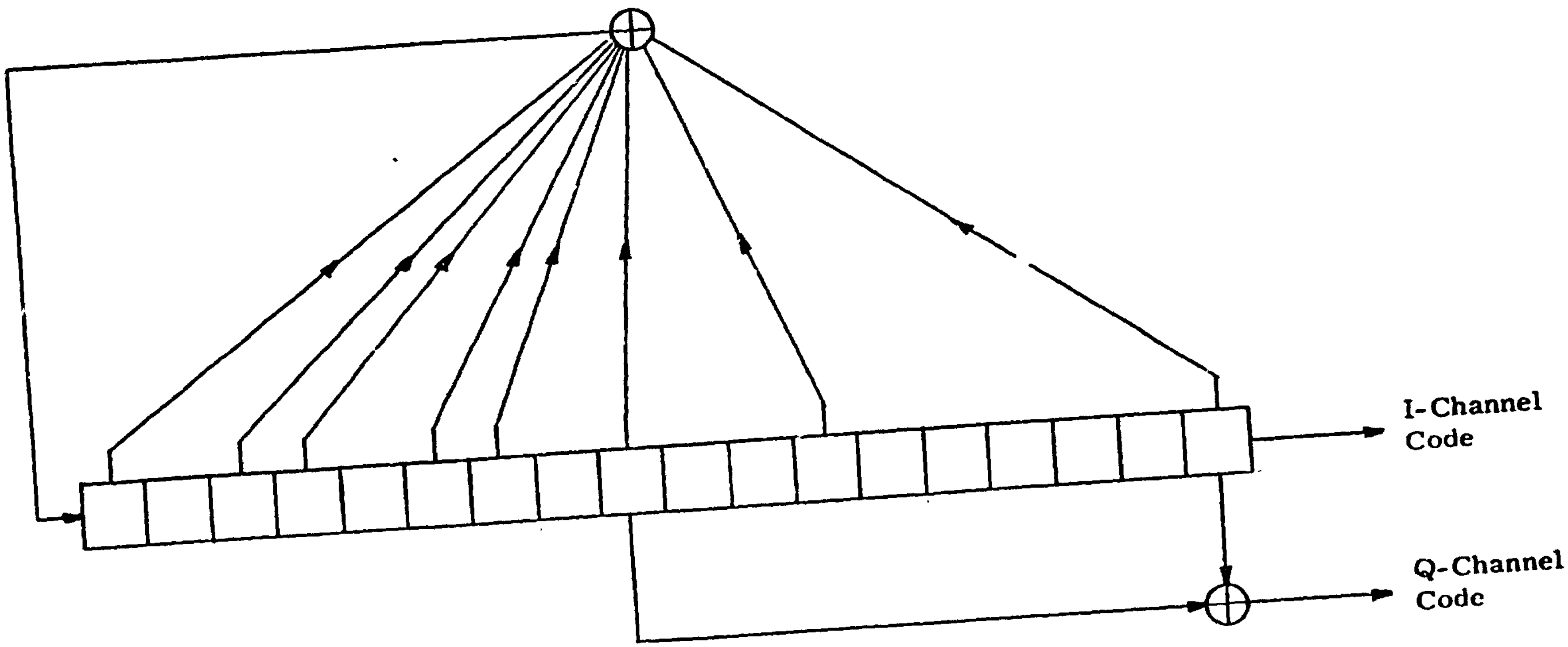

Figure 3. Mode 1 Return Link Code Generator for User Code Assignment #12

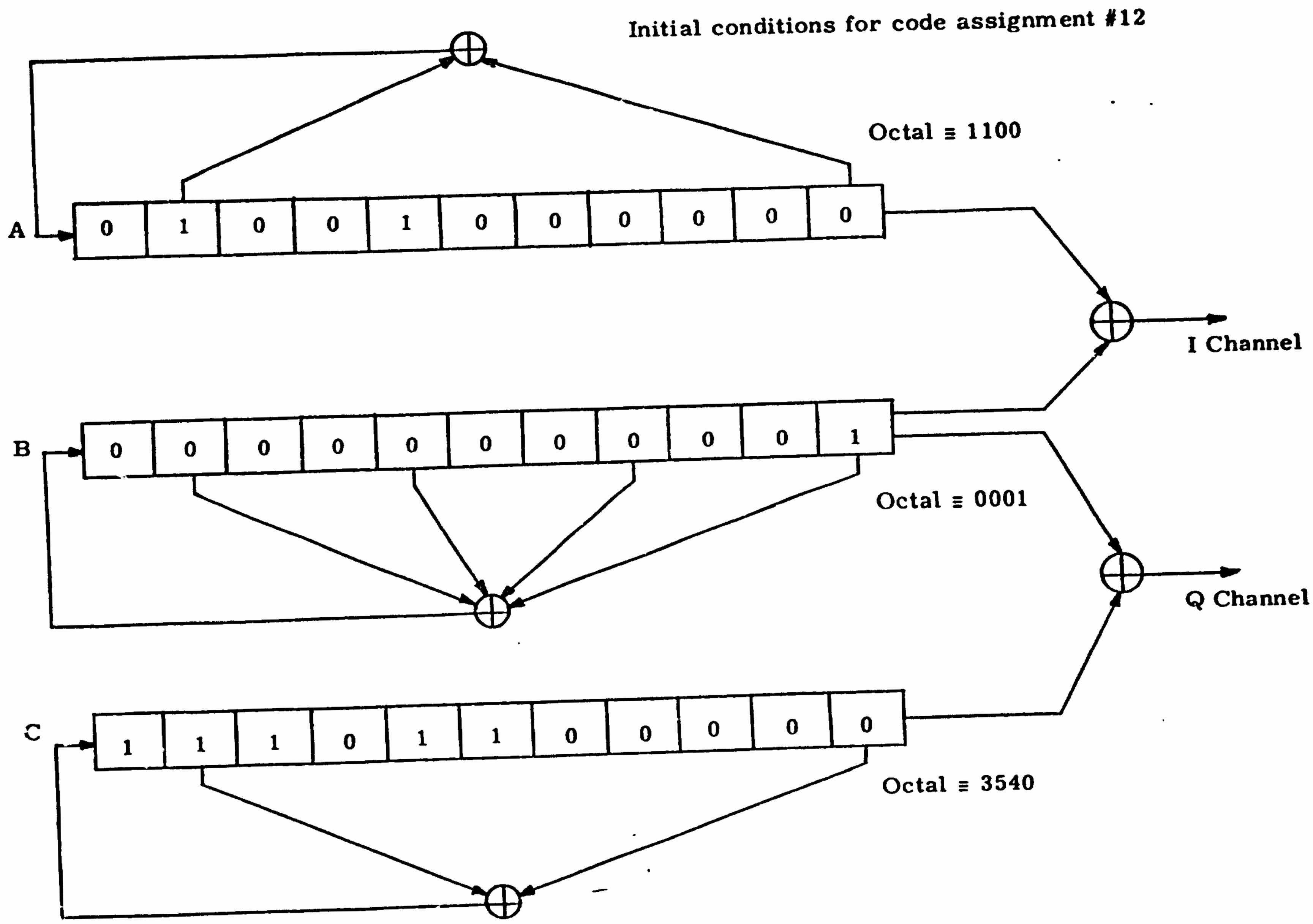

Figure 4. Generator for Mode 2 Return Link Codes

Table 1. Summary of TDRSS Code Libraries

1	2	3	4	5	6	7
User Number	Forward Link Command Channel Initial Conditions Register A	Forward Link Range Channel Feedback Taps	Mode 1 Return Link Feedback Taps	Mode 1 Return Link Channel Phase Difference in Chips	Mode 2 Return Link Initial Conditions Register A	Mode 2 Return Link Initial Conditio Register C
1	1010011	1022005	1000047	94672	0004	0006
2	0110011	1022027	1431503	35529	0040	0060
3	1100000	1022055	1012633	100097	0100	0140
4	0010000	1022131	1012715	37927	0200	0300
5	1010000	1022145	1010551	20125	0240	0360
6	0110000	1022225	1101063	127909	1004	3406
7	1110000	1022311	1010463	124686	2010	3014
8	0001000	1022443	1002031	25966	0104	0146
9	1001000	1022461	1010313	122601	2522	3773
10	0101000	1022621	1125611	130803	1244	3766
11	1001011	1023045	1002061	118080	2440	3660
12	0011000	1023103	1011333	121824	1100	2540
13	0101011	1023111	1010163	81188	3203	2702
14	0011011	1023221	1002133	44099	0071	2045
15	1111000	1023405	1010133	36812	0162	0113
16	1011011	1024017	1011553	66194	0344	0226
17	0111011	1024027	1011347	53036	0710	0454
18	0100100	1024063	1116115	74458	1621	1131
19	1111011	1024065	1011261	104408	0335	2263
20	0000111	1024305	1011571	1[illegible]2103	0672	0547
21	1010100	1025105	1012527	97306	1565	1317
22	1000111	1025141	1012547	82029	3535	2363
23	0100111	1026023	1007705	88320	3272	0747
24	0001100	1026043	1012363	106369	1650	3174
25	1001100	1026061	1013525	43037	2050	3074

Table 1. (continued)

1	2	3	4	5	6	7
User Number	Forward Link Command Channel Initial Conditions Register A	Forward Link Range Channel Feedback Taps	Mode 1 Return Link Feedback Taps	Mode 1 Return Link Channel Phase Difference in Chips	Mode 2 Return Link Initial Conditions Register A	Mode 2 Return Link Initial Conditior Register C
26	0101100	1030145	1201011	38185	2422	3633
27	0010111	1030161	1007543	117247	1044	3466
28	0011100	1030215	1002171	77081	2110	3154
29	1011100	1030303	1007501	46045	0425	2637
30	0111100	1030311	1002211	127440	1053	1476
31	0110111	1030321	1007417	91623	0532	0767
32	0000010	1030341	1716201	124677	2047	1064
33	1000010	1030407	1007315	124514	0117	2150
34	0100010	1034013	1015037	25407	1171	1505
35	1100010	1034051	1002441	87008	3471	2245
36	1110111	1034105	1230121	53510	3457	2270
37	1010010	1035021	1014365	38428	3136	0561
38	0001111	1040043	1014475	33738	3671	2145
39	1110010	1040051	1007165	45941	3562	0313
40	0001010	1040117	1141703	30125	2225	1337
41	1001010	1040205	1002623	26908	2566	3715
42	0101111	1040247	1021553	26391	3025	[illegible]37
43	1101010	1040361	1002741	61389	1271	1745
44	0011010	1040463	1020277	22264	2562	3713
45	1101111	1040465	1001661	59945	1344	3626
46	0011111	1040545	1017611	113772	0135	2163
47	1011111	1040645	1003011	116324	2722	3473
48	1111111	1040721	1017511	103124	2466	3655
49	1000110	1041011	1001631	65712	0661	2551
50	0110101	1041035	1017311	58281	1543	1322

Table 1. (continued)

1	2	3	4	5	6	7
User Number	Forward Link Command Channel Initial Conditions Register A	Forward Link Range Channel Feedback Taps	Mode 1 Return Link Feedback Taps	Mode 1 Return Link Channel Phase Difference in Chips	Mode 2 Return Link Initial Conditions Register A	Mode 2 Return Link Initial Condition Register C
51	1110101	1041207	1003035	104186	3307	2644
52	0010110	1041225	1003053	56674	2617	1510
53	0001101	1041423	1003451	64366	1436	3221
54	0110110	1041445	1017071	25412	3566	0315
55	1101101	1041451	1001567	130143	3354	0632
56	0001110	1041505	1003461	100841	2731	1465
57	0011101	1200441	1021473	95661	3125	2577
58	0101110	1640441	1021475	111502	1257	1770
59	1101110	1320441	1003521	112683	2740	3420
60	1011101	1150441	1133015	74581	1371	1605
61	1011110	1230441	1001427	130092	2762	3413
62	0111110	1244441	1004073	129297	3622	0133
63	1111110	1114441	1001361	94465	3444	0266
64	0000001	1422441	1402335	92879	3110	0554
65	1000001	1062441	1001253	39646	2221	1331
66	1111101	1046441	1301323	25494	0443	2662
67	1100001	1221441	1001165	128965	0732	0467
68	0010001	1411441	1015631	115634	1665	1157
69	0000011	1111441	1001141	25409	3553	2336
70	0110001	1045441	1025051	78323	1532	3367
71	1110001	1203441	1001023	103808	3265	2757
72	0001001	1700241	1021355	42612	2553	1736
73	1000011	1640241	1010615	98036	1326	3675
74	0100011	1460241	1021363	27778	3261	2751
75	1101001	1260241	1000757	77032	3061	2451

Table 1. (continued)

1	2	3	4	5	6	7
User Number	Forward Link Command Channel Initial Conditions Register A	Forward Link Range Channel Feedback Taps	Mode 1 Return Link Feedback Taps	Mode 1 Return Link Channel Phase Difference in Chips	Mode 2 Return Link Initial Conditions Register A	Mode 2 Return Link Initial Conditior Register C
76	0011001	1214241	1060065	110203	2143	1122
77	1011001	1211241	1010741	75695	0357	2230
78	0111001	1031241	1205651	98275	0736	0461
79	1111001	1440641	1004163	70757	1732	3067
80	0000101	1420641	1004205	117105	2513	1756
81	1000101	1060641	1000621	87745	1226	3735
82	1100011	1230141	1530521	125022	1414	3212
83	1100101	1070141	1000517	121793	2347	1224
84	0010011	1304141	1021273	69616	3176	0501
85	1010101	1414141	1000407	61200	3757	2030

3.0 SPURIOUS CODES DUE TO FILTERING AND LIMITING

3.1 Introduction

The filtering and hard-limiting of a staggered quadriphase PN (SQPN) signal can result in spurious PN sequences in the I and Q channels. These spurious sequences may result in false correlation peaks whose amplitude and phase will depend on the codes originally used on the I and Q channels. This phenomenon was reported in [1] where a physical explanation for its occurrence was described. In the section, we describe techniques for the construction of TDRS code libraries for which this type of interference is avoided.

3.2 Analytical Description of the Phenomena of Spurious Code Generation

In this section, we prove the following:

Result: Suppose that a and b are binary (±1) sequences on the I and Q channels, respectively, and that sequence b is delayed 1/2 bit (SQPN). The spurious sequences which are generated on the I and Q channels, respectively, due to filtering and hard-limiting are:

$$\text{I Channel} \equiv a \cdot b \cdot b_1$$

$$\text{Q Channel} \equiv b \cdot a_{-1} \cdot a \, ,$$

where b_1 is the sequence b delayed by one chip and a_{-1} is the sequence a advanced by one chip.

Proof: The sequence a on the I channel gets amplitude modulation due to filtering and hard-limiting when and only when the transition sequence of b, namely $b \cdot b_1$, is -1. The output a_{fL} of the I channel after filtering and hard-limiting may thus be computed as

$$a_{fL} = a\left[1 + m\left(\frac{1-(b\cdot b_1)}{2}\right)\right]$$

$$= a\left[1 + m'(1 - b \cdot b_1)\right]$$

$$a_{fL} = a(1+m') - m' a \cdot b \cdot b_1 .$$

The spurious sequence out of the I channel after filtering and hard-limiting is $a \cdot b \cdot b_1$.

Similarly, the sequence b on the Q channel gets amplitude modulation due to filtering and hard-limiting when and only when the transition sequence of a, namely $(a \cdot a_1)_{-1}$, is -1. The transition sequence in this case must be advanced one chip because of the staggering of the sequence b. The output b_{fL} of the Q channel after filtering and hard-limiting may thus be computed as:

$$b_{fL} = b\left[1 + m'\left(\frac{1 - (a \cdot a_1)_{-1}}{2}\right)\right]$$

$$= b\left[1 + m'\left(1 - (a \cdot a_1)_{-1}\right)\right]$$

$$= b(1+m') - m' b \cdot (a \cdot a_1)_{-1} .$$

The spurious sequence out of the Q channel after filtering and hard-limiting is $b \cdot a_{-1} \cdot a$.

3.3 Results for Maximal Sequences

If the sequences a and b on the I and Q channels, respectively, are maximal PN sequences, then the formulas given in the above paragraph for the spurious sequences on the I and Q channels become:

$$\text{I Channel} \equiv a \cdot b_{\phi_b(1)}$$

$$\text{Q Channel} \equiv b \cdot a_{-1+\phi_a(1)} ,$$

where ϕ_a and ϕ_b are the shift-and-add functions of the maximal sequences a and b, respectively.

If the sequence b is a proper phase shift $b = a_\tau$ of the maximal sequence a, as in the case of the Mode 1 Return link codes, then the above formulas for the spurious sequences become:

$$\text{I Channel:} \quad a_{\phi_a\left(\tau + \phi_a(1)\right)}$$

$$\text{Q Channel:} \quad a_{\tau + \phi_a\left(-1 - \tau + \phi_a(1)\right)} .$$

3.3.1 Selection of the Mode 1 Return Link Code Library

The Mode 1 Return Link codes were selected from the 7776 maximal PN sequences of period $2^{18}-1$. The following three criteria were imposed on those sequences which were selected for the TDRS Mode 1 return link library.

(a) The number of feedback taps required by the shift register generating the sequence is at most 8.

(b) The modulo-2 sum of the output of the ninth and eighteenth stages of the shift register, which is used as the Q channel code, differs in phase from the output of the eighteenth stage, which is used as the I channel code. Table 2 contains a list of 128 codes meeting criteria (a) and (b). Column 1 of Table 2 contains the feedback taps for the shift register generators in octal notation. Column 2 of Table 2 contains the minimum absolute value of the phase differences between the two codes.

(c) The maximal code and its phase shift, which are used on the I channel and Q channel, respectively, result in a spurious code.

The spurious codes which appear on the I and Q channels are both phase shifts of the original codes on these channels. The codes chosen for the TDRS library were such that each of the spurious codes differs from the original codes by at least 5000 chips. The formula for each of the spurious codes was given above in terms of the original code a used for the I channel, its phase shift a_τ used for the Q channel, and the shift-and-add function of the code.

For each code selected, the following inequalities were satisfied:

$$A = \left|\phi_a\left(\tau + \phi_a(1)\right)\right| > 5000$$

$$B = \left|\tau + \phi_a\left(-1 - \tau + \phi_a(1)\right)\right| > 5000$$

$$C = \left|\tau - \phi_a\left(\tau + \phi_a(1)\right)\right| > 5000$$

$$D = \left|\phi_a\left(-1 - \tau + \phi_a(1)\right)\right| > 5000$$

The exact values for these quantities which represent the separation between each of the spurious codes and the I and Q channel codes are give in columns 3 through 6 of Table 2.

Table 2. Spurious Code Separation on I and Q Channels for Maximal PN Codes

Feedback Tap	τ	A	B	C	D
1000047	94672	99071	46572	68400	120899
1431503	35529	73642	10990	38113	46519
1012633	100097	71591	10792	90455	110889
1012715	37927	36653	100986	74580	123230
1010551	20125	70588	20535	90713	40660
1101063	127909	121432	40600	12802	87309
1010463	124686	71164	105604	53522	31853
1011267	48528	100201	2177	113414	46351
1010313	122601	97594	29931	25007	109611
1125611	130803	110527	94362	20813	36441
1010211	42628	4092	104344	46720	115171
1011333	121824	32249	42227	108070	79597
1010163	81188	49233	42378	31955	38810
1011533	61209	120461	126259	80473	74675
1010133	36812	124301	51122	101030	14310
1011553	66194	66314	27402	120	38792
1011347	53036	28176	28342	81212	81378
1116115	74458	116998	67722	42540	6736
1011261	104408	117487	87531	13079	70204
1011571	102103	118005	42205	15902	117835
1012527	97306	99310	74714	65527	22592
1012547	82029	7646	31783	74383	50246
1007705	88320	127211	46834	38891	126989
1012363	106369	77503	12219	78271	118588
1013525	43037	106473	69532	112633	26495
1201011	38185	35745	66444	73930	28259
1007543	117274	112079	63956	32817	80940
1013625	98985	2934	69777	101919	93381
1007501	46045	79330	71959	125375	25914
1014555	49892	52171	127574	2279	84677
1007417	91623	61619	85635	30004	84885
1716201	124677	112743	107348	11934	30118
1007315	124514	118034	97016	6480	40613
1015037	25407	72334	63683	46927	38276
1007263	79938	84748	112181	4810	70024
1230121	53510	10542	15851	64052	37659
1014365	38428	11648	29760	26780	68188
1014475	33738	14811	8893	20927	42631
1007165	45941	44146	22232	90087	23709
1141703	30125	124973	91583	94848	61458
1007121	57531	113575	57062	56044	469

Table 2 (continued)

Feedback Tap	T	A	B	C	D
1021553	26391	108868	117098	126884	118654
1001705	83753	1156	3456	84909	87209
1020277	22264	78516	49064	100780	71328
1001661	59945	18884	29994	41061	89939
1017611	113772	34797	50703	78975	63069
1001651	106789	106999	34083	210	121271
1017511	103124	78712	46784	80307	112235
1001631	65712	116272	77798	50560	12086
1017311	58281	127059	116337	76803	58056
1001625	112199	303	128769	112502	21175
1017161	123092	122573	99984	519	39067
1001607	77447	288	72712	77159	4735
1017071	25412	44340	119670	18928	117061
1001567	130143	107134	40714	23009	89429
1016705	95338	109774	3869	57031	91469
1021473	95661	130065	13754	36417	81907
1021475	111502	74499	61587	37003	89054
1001453	62504	3834	57152	58670	119656
1133015	74581	38260	84248	112841	103314
1001427	130092	52159	111397	79892	18695
1101533	91020	92236	45479	1216	45541
1001361	94465	130782	55137	36317	112541
1402335	92879	87112	78947	5767	13932
1001253	39646	80534	74193	120180	34547
1301323	25494	71738	85506	97232	111000
1001165	128965	42198	63897	90980	65068
1015631	115634	113704	33036	32805	82598
1001141	25409	29125	120390	54534	127799
1025051	78323	88542	79437	95278	104383
1001023	103808	69049	18623	34759	122431
1021355	42612	16253	106684	58865	64072
1010615	98036	129971	130493	34136	33614
1021363	27778	89845	116555	117623	88777
1000757	77032	86262	26119	9230	50913
1060065	110203	76345	58589	33858	93451
1010741	75695	48672	129881	124367	54186
1205651	98275	14939	109980	113214	11705
1000743	39070	10743	35679	49813	3391
1021237	25929	60652	4856	34723	30785
1000621	87745	46969	39633	127429	48112
1530521	125022	102429	117511	22593	19610
1000517	121793	114231	37410	26119	102940
1021273	69616	57251	59068	12365	128684
1000407	61200	97605	129487	103338	71456

Table 2 (continued)

Feedback Tap	T	A	B	C	D
1324243	80029	120933	104897	22894	24858
1000355	86513	26326	98844	60187	76786
1020753	60361	63651	415	3290	59946
1000347	56667	60953	69200	4286	12533
1020771	105837	78433	77517	77873	78789
1000333	47180	87301	107958	40121	60778
1032067	25351	55549	91916	80900	117267
1000201	131066	65543	105253	65534	25813
1017243	55015	50209	116896	105224	61881
1000173	44889	63689	104930	18800	60041
1017261	27451	123602	2029	111090	25422
1011055	122484	83516	15647	56143	106837
1016435	111283	79067	108352	32216	2931
1000077	50633	22399	90459	73032	121051
1016561	74848	124695	13476	62600	88324
1000743	39070	10743	35679	49813	3391
1002031	25966	89999	18895	115965	7071
1002061	118080	29509	34328	88571	83752
1002075	53402	1541	27973	51861	25429
1002133	44099	18763	21782	25336	65881
1002171	77081	18733	65585	95814	119477
1002211	127440	73780	101224	60923	33479
1002441	87008	117880	121608	30872	34600
1002623	26908	33839	56636	60747	83544
1002705	123686	81303	122913	42383	773
1002741	61389	35562	37406	96951	23983
1003011	116324	123251	61815	6927	54509
1003035	104186	11708	79811	115894	24315
1003053	56674	88670	47747	116799	104421
1003451	64366	105487	35997	92290	100363
1003461	100841	22023	62563	122864	38278
1003521	112683	17377	74519	130060	38164
1004073	129297	81429	13735	47868	119111
1004163	70757	35315	93182	35442	98204
1004205	117105	53576	19897	63529	125141
1004313	105605	33792	122434	122746	16829
1004455	73891	18375	69091	55516	4800
1004545	61333	59745	52380	121078	8953
1004623	35112	118949	47688	83837	12576
1004643	24800	4063	30665	28863	5865
1004645	98791	43137	108322	55654	9531
1004711	96374	9510	117598	105884	21224
1005035	70603	18936	86087	89539	15434

3.4 Results for Gold Codes

When two members of a family of Gold codes are used in SQPN, then the spurious sequences which appear on the I and Q channel due to filtering and hard-limiting are also members of the Gold family. More precisely, we have the following.

Result: Let $\{g^i \equiv a \cdot b_i\}$ be a family of Gold codes generated by the maximal linear PN sequences a and b. Suppose that, on the I channel, we have the code g^i and that on the Q channel, we have the code g^j. The spurious codes on the I and Q channels, respectively, are

$$\text{I Channel:} \quad g^{i-1+\phi_b[j-i+\phi_b(1)]}$$

$$\text{Q Channel:} \quad g^{j-1+\phi_b[-(j-i)-1+\phi_b(1)]}.$$

Proof: Using the result of Section 3.2 with $a = g^i$ and $b = g^j$, we have the following expression for the spurious codes:

$$\begin{aligned}
\text{I Channel:} \quad & a \cdot b \cdot b_1 \\
& g^i \cdot g^j \cdot (g^j)_1 \\
& a \cdot b_i \cdot a \cdot b_j \cdot a_1 \cdot b_{j+1} \\
& a_1 \cdot b_i \cdot (b_j \cdot b_{j+1}) \\
& a_1 \cdot \left(b_i \cdot b_{j+\phi_b(1)}\right) \\
& a_1 \cdot b_{i+\phi_b(j-i+\phi_b(1))} \\
& g^{i-1+\phi_b(j-i+\phi_b(1))}
\end{aligned}$$

Q Channel: $b \cdot a_{-1} \cdot a$

$$g^j \cdot (g^i)_{-1} (g^i)$$

$$a \cdot b_j \cdot a_{-1} \cdot b_{i-1} \cdot a \cdot b_i$$

$$a_{-1} \cdot b_j \cdot (b_{i-1} \cdot b_i)$$

$$a_{-1} \cdot b_j \cdot b_{i-1+\phi_b(1)}$$

$$a_{-1} \cdot b_{j+\phi_b\left((i-j)-1+\phi_b(1)\right)}$$

$$g^{j-1+\phi_b\left((i-j)-1+\phi_b(1)\right)}.$$

Corollary: If the argument of the shift-and-add function in the above expressions is zero, then the spurious sequence in the corresponding channel is a maximal sequence.

3.5 Construction of Mode 2 Return Link Library

The code library for the Mode 2 return link consists of 85 code pairs of period $(2^{11}-1)$ selected from the members of a Gold family. The first member of the code pair is used for the I channel and the second member of the code pair is used for the Q channel. The code pairs of the library are chosen so that the spurious Gold codes which occur on the I and Q channels, respectively, due to filtering and hard-limiting do not belong to the library of 85 codes and hence cannot cause interference problems. In the following section, we show how the library is constructed.

3.5.1 Code Library Equations

The spurious codes which appear on the I and Q channel, respectively, when the code pair of the library is g^i and g^j have been shown in Section 3.4 to be:

I Channel Spurious Code: $g^{i-1+\phi_b[(j-i)+\phi_b(1)]}$

Q Channel Spurious Code: $g^{j-1+\phi_b[-(j-i)-1+\phi_b(1)]}$.

We first choose j and i such that the argument of ϕ_b in the expression for the Q channel spurious code is zero. Thus, we have:

$$j = i - 1 + \phi_b(1) .$$

$\phi_b(1)$ is computed to be 1029 and hence we have:

$$\boxed{j = i + 1028} .$$

This choice of j guarantees that the spurious code which appears on the Q channel will be some phase shift of the maximal sequence a and hence not a member of the code library.

With the above choice of j, the spurious code which appears on the I channel is

$$g^{i-1+\phi_b[2\phi_b(1)-1]} = g^{i-1+\phi_b(10)} .$$

$\phi_b(10)$ is computed to be 1495 and hence we have for the spurious Gold code on the I channel:

$$g^{i+1494} .$$

We now choose for the codes of the I channel the Gold codes

$$g^0, g^1, \ldots, g^{465}$$

and for the corresponding codes of the Q channel

$$g^{1028}, g^{1029}, \ldots, g^{1493} .$$

The spurious codes on the Q channel will always be a phase shift of the maximal sequence a. The spurious codes on the I channel will be the Gold codes

$$g^{1494}, g^{1495}, \ldots, g^{1959} .$$

3.5.2 Generation of Gold Codes

In order to generate the Gold codes specified in Section 3.5.1, we define a to be the maximal linear code generated by the polynomial $1+x^2+x^5+x^8+x^{11}$ and having the initial conditions

$$0001 \text{ (octal)} = 00000000001 \text{ (binary)} .$$

We define b to be the maximal linear code generated by the polynomial $1+x^2+x^{11}$ and having the initial conditions

$$0001 \text{ (octal)} = 00000000001 \text{ (binary)}.$$

The Gold code $g^0 = a \cdot b$, for example, is thus generated by the shift register configuration shown in Figure 5.

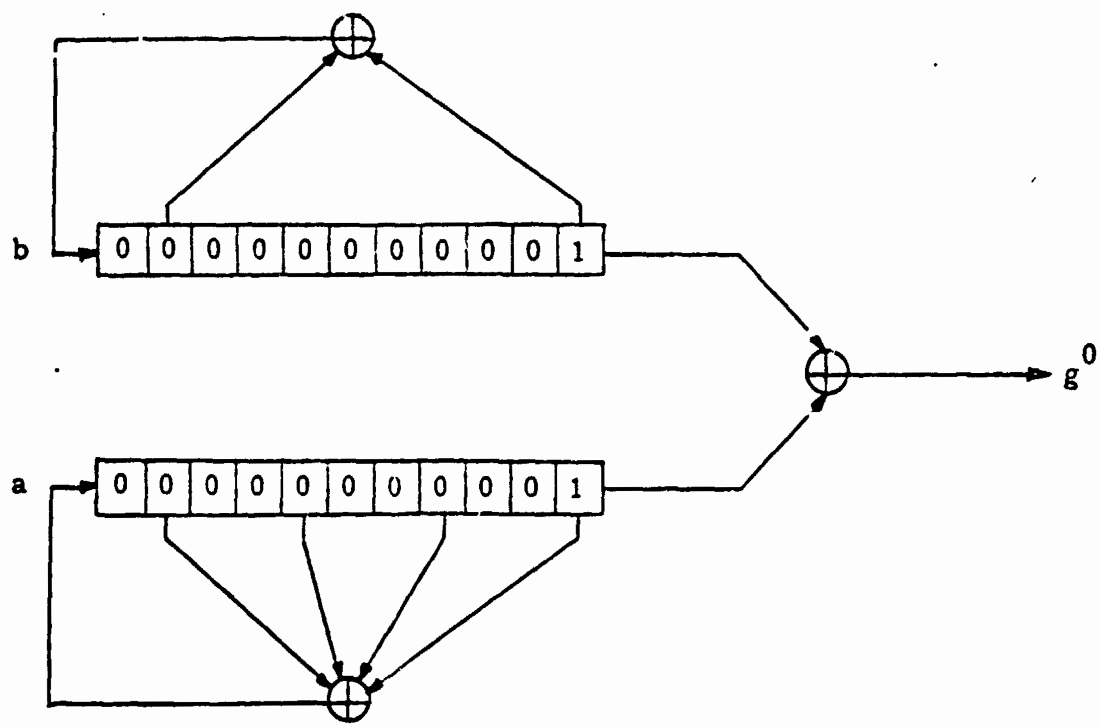

Figure 5. Generation of Gold Codes

The Gold code g^k is generated by delaying the sequence b by k bits with respect to the sequence a and multiplying the resultant sequences a and b_k to obtain $g^k = a \cdot b_k$.

To obtain the initial conditions of the sequence b_1, we clock the generating shift register in Figure 5 $(2^{11}-1)-1$ times.

The initial conditions of the sequence b_k correspond to the state vector of the upper register of Figure 5 when it is clocked $(2^{11}-1)-k$ times. Thus, to generate the Gold code g^k using the configuration of Figure 5, we use these initial conditions for the upper register and the initial conditions 0001 (octal) ≡ 00000000001 (binary) for the lower register. The initial conditions required to generate the code pairs determined by the equations of Section 3.5.1 have been computed and are listed in Table 3.

Table 3 contains a listing of the 466 pairs of Gold codes (specified by the initial conditions of the registers A and C in Figure 4) determined by the equations of Section 3.5.1. Each of the 466 code pairs in the listing has the property that, if the first member is used on the I channel and the second member is used on the Q channel, the spurious codes which result from filtering and hard-limiting will not be any of the codes contained in this listing. Those code pairs of Table 3 for which both members are balanced are designated by * and the first 85 such code pairs were taken for the TDRSS Mode 2 return link code library.

Table 3. Gold Code Pairs for I and Q Channels

Pair	Initial Conditions for I Channel	Initial Conditions for Q Channel	Number of Ones for I	Number of Ones for Q	Number of Balanced Pairs
1*	0001	2001	992	1024	0
2*	0002	0003	1024	992	0
3	0004	0006	1024	1024	1
4*	0010	0014	992	992	1
5*	0020	0030	1024	1056	1
6	0040	0060	1024	1024	2
7	0100	0140	1024	1024	3
8	0200	0300	1024	1024	4
9*	0400	0600	992	1056	4
10*	1001	1401	1056	1024	4
11*	2002	3003	992	1024	4
12*	0005	2007	992	1024	4
13*	0012	0017	1056	1024	4
14*	0024	0036	1024	992	4
15*	0050	0074	1056	992	4
16*	0120	0170	1024	992	4
17	0240	0360	1024	1024	5
18*	0500	0740	992	1056	5
19*	1201	1701	992	1024	5
20*	2402	3603	1024	992	5
21	1004	3406	1024	1024	6
22	2010	3014	1024	1024	7
23*	0021	2031	992	992	7
24*	0042	0063	1024	992	7
25	0104	0146	1024	1024	8
26*	0210	0314	992	992	8
27*	0420	0630	1056	1024	8
28*	1041	1461	992	1024	8
29*	2102	3143	1056	1024	8
30*	0205	2307	992	1024	8
31*	0412	0617	1024	1056	8
32*	1025	1437	1056	1056	8
33*	2052	3077	1024	992	8

34*	0125	2177	992	992	8
35*	0252	0377	1056	1024	8
36*	0524	0776	1056	1024	8
37*	1251	1775	1024	1056	8
38	2522	3773	1024	1024	9
39	1244	3766	1024	1024	10
40*	2510	3754	992	1056	10
41*	1220	3730	1024	992	10
42	2440	3660	1024	1024	11
43	1100	3540	1024	1024	12
44*	2200	3300	992	992	12
45*	0401	2601	1024	992	12
46*	1003	1402	992	992	12
47*	2006	3005	992	1024	12
48*	0015	2013	1024	992	12
49*	0032	0027	992	992	12
50*	0064	0056	1024	1056	12
51*	0150	0134	992	1056	12
52*	0320	0270	1024	992	12
53*	0640	0560	992	1056	12
54*	1501	1341	1024	992	12
55	3203	2702	1024	1024	13
56*	2407	1604	992	1056	13
57*	1016	3411	992	1024	13
58*	2034	3022	1024	992	13
59	0071	2045	1024	1024	14
60	0162	0113	1024	1024	15
61	0344	0226	1024	1024	16
62	0710	0454	1024	1024	17
63	1621	1131	1024	1024	18
64*	3443	2262	992	992	18
65*	3106	0545	1056	1024	18
66*	2215	1313	1056	1024	18
67*	0433	2626	992	1024	18
68*	1067	1454	1056	1024	18
69*	2156	3131	1024	992	18
70	0335	2263	1024	1024	19
71	0672	0547	1024	1024	20
72	1565	1317	1024	1024	21
73*	3353	2636	992	1056	21
74*	2727	1474	1056	1024	21
75*	1056	3171	1024	992	21

76	3535	2363	1024	1024	22
77	3272	0747	1024	1024	23
78*	2565	1717	1056	1056	23
79*	1352	3637	992	1024	23
80*	2724	3476	1024	1056	23
81	1650	3174	1024	1024	24
82*	3521	2371	1056	1056	24
83*	3242	0763	1056	1024	24
84*	2505	1747	1056	1024	24
85*	1212	3717	992	1024	24
86*	2424	3636	1024	1056	24
87*	1050	3474	992	1056	24
88*	2120	3170	1056	1024	24
89*	0241	2361	992	1024	24
90*	0502	0743	1056	1024	24
91*	1205	1707	1056	1024	24
92*	2412	3617	992	1024	24
93*	1024	3436	1024	992	24
94	2050	3074	1024	1024	25
95*	0121	2171	992	1056	25
96*	0242	0363	1024	992	25
97*	0504	0746	992	1056	25
98*	1211	1715	1024	992	25
99	2422	3633	1024	1024	26
100	1044	3466	1024	1024	27
101	2110	3154	1024	1024	28
102*	0221	2331	992	992	28
103*	0442	0663	1056	1024	28
104*	1105	1547	1056	1024	28
105*	2212	3317	1024	1056	28
106	0425	2637	1024	1024	29
107	1053	1476	1024	1024	30
108*	2126	3175	992	992	30
109*	0255	2373	1024	1056	30
110	0532	0767	1024	1024	31
111*	1265	1757	992	992	31
112*	2552	3737	1056	1024	31
113*	1324	3676	1024	1056	31
114*	2650	3574	1056	992	31
115*	1520	3370	1056	1024	31
116*	3241	2761	1024	1056	31
117*	2503	1742	1056	1056	31

118*	1206	3705	1024	992	31
119*	2414	3612	1056	992	31
120*	1030	3424	1056	1024	31
121*	2060	3050	1056	1024	31
122*	0141	2121	992	1024	31
123*	0302	0243	1024	992	31
124*	0604	0506	1056	1056	31
125*	1411	1215	992	1024	31
126*	3023	2432	1024	1056	31
127	2047	1064	1024	1024	32
128	0117	2150	1024	1024	33
129*	0236	0321	992	992	33
130*	0474	0642	1024	1056	33
131	1171	1505	1024	1024	34
132*	2362	3213	992	992	34
133*	0745	2427	1024	992	34
134*	1713	1056	992	992	34
135*	3627	2134	992	1024	34
136*	3456	0271	992	1024	34
137*	3134	0562	1024	992	34
138*	2271	1345	1056	992	34
139*	0563	2712	1024	1056	34
140*	1347	1624	1056	1056	34
141*	2716	3451	1056	1024	34
142*	1634	3122	1024	992	34
143	3471	2245	1024	1024	35
144*	3162	0513	992	992	35
145*	2345	1227	1024	1056	35
146*	0713	2456	1056	992	35
147*	1627	1134	1024	1056	35
148	3457	2270	1024	1024	36
149	3136	0561	1024	1024	37
150*	2275	1343	992	1056	37
151*	0573	2706	992	1024	37
152*	1367	1614	1056	1024	37
153*	2756	3431	1056	1024	37
154*	1734	3062	1024	1056	37
155	3671	2145	1024	1024	38
156	3562	0313	1024	1024	39
157*	3344	0626	992	1056	39
158*	2711	1455	1024	992	39
159*	1622	3133	992	1056	39

160*	3445	2267	1056	1024	39
161*	3112	0557	1024	992	39
162	2225	1337	1024	1024	40
163*	0453	2676	992	1056	40
164*	1127	1574	1056	1024	40
165*	2256	3371	1024	1056	40
166*	0535	2763	992	1056	40
167*	1273	1746	1024	1056	40
168	2566	3715	1024	1024	41
169*	1354	3632	992	992	41
170*	2730	3464	1056	1024	41
171*	1660	3150	992	1024	41
172*	3541	2321	992	1024	41
173*	3302	0643	992	1024	41
174*	2605	1507	992	1024	41
175*	1412	3217	1024	1056	41
176	3025	2437	1024	1024	42
177*	2053	1076	992	1056	42
178*	0127	2174	992	1024	42
179*	0256	0371	992	1024	42
180*	0534	0762	1024	992	42
181	1271	1745	1024	1024	43
182	2562	3713	1024	1024	44
183	1344	3626	1024	1024	45
184*	2710	3454	992	992	45
185*	1620	3130	992	1024	45
186*	3441	2261	1056	1024	45
187*	3102	0543	992	1024	45
188*	2205	1307	1024	1056	45
189*	0413	2616	992	992	45
190*	1027	1434	992	1024	45
191*	2056	3071	1024	1056	45
192	0135	2163	1024	1024	46
193*	0272	0347	1056	992	46
194*	0564	0716	992	1024	46
195*	1351	1635	1024	992	46
196	2722	3473	1024	1024	47
197*	1644	3166	1056	992	47
198*	3511	2355	1024	1056	47
199*	3222	0733	992	1056	47
200*	2445	1667	992	1024	47
201*	1112	3557	992	1024	47

202*	2224	3336	1056	1024	47
203*	0451	2675	992	1024	47
204*	1123	1572	992	1024	47
205*	2246	3365	1056	1024	47
206*	0515	2753	992	1024	47
207*	1233	1726	1024	992	47
208	2466	3655	1024	1024	48
209*	1154	3532	992	1056	48
210*	2330	3264	1024	992	48
211	0661	2551	1024	1024	49
212	1543	1322	1024	1024	50
213	3307	2644	1024	1024	51
214	2617	1510	1024	1024	52
215	1436	3221	1024	1024	53
216*	3075	2443	1056	1056	53
217*	2173	1106	1056	1024	53
218*	0367	2214	1056	1024	53
219*	0756	0431	1024	992	53
220*	1735	1063	1056	1056	53
221*	3673	2146	1024	1056	53
222	3566	0315	1024	1024	54
223	3354	0632	1024	1024	55
224	2731	1465	1024	1024	56
225*	1662	3153	992	1056	56
226*	3545	2327	992	1024	56
227*	3312	0657	1024	1056	56
228*	2625	1537	992	992	56
229*	1452	3277	1024	1056	56
230	3125	2577	1024	1024	57
231*	2253	1376	992	1056	57
232*	0527	2774	1024	992	57
233	1257	1770	1024	1024	58
234*	2536	3761	992	992	58
235*	1274	3742	1056	1024	58
236*	2570	3704	992	1024	58
237*	1360	3610	1024	992	58
238	2740	3420	1024	1024	59
239*	1700	3040	1056	992	59
240*	3601	2101	1056	1024	59
241*	3402	0203	1024	992	59
242*	3004	0406	1056	1056	59
243*	2011	1015	992	1024	59

244*	0023	2032	992	1024	59
245*	0046	0065	1024	1056	59
246*	0114	0152	992	1056	59
247*	0230	0324	1056	1024	59
248*	0460	0650	992	1024	59
249*	1141	1521	992	1024	59
250*	2302	3243	992	1024	59
251*	0605	2507	1024	1056	59
252*	1413	1216	1056	1056	59
253*	3027	2434	1024	992	59
254*	2057	1070	1056	1056	59
255*	0137	2160	1024	1056	59
256*	0276	0341	992	1056	59
257*	0574	0702	1024	992	59
258	1371	1605	1024	1024	60
259	2762	3413	1024	1024	61
260*	1744	3026	1056	1056	61
261*	3711	2055	1024	1056	61
262	3622	0133	1024	1024	62
263	3444	0266	1024	1024	63
264	3110	0554	1024	1024	64
265	2221	1331	1024	1024	65
266	0443	2662	1024	1024	66
267*	1107	1544	992	992	66
268*	2216	3311	1024	1056	66
269*	0435	2623	992	992	66
270*	1073	1446	1024	992	66
271*	2166	3115	1056	992	66
272*	0355	2233	1024	1056	66
273	0732	0467	1024	1024	67
274	1665	1157	1024	1024	68
275	3553	2336	1024	1024	69
276*	3326	0675	1056	992	69
277*	2655	1573	1024	1056	69
278	1532	3367	1024	1024	70
279	3265	2757	1024	1024	71
280	2553	1736	1024	1024	72
281	1326	3675	1024	1024	73
282*	2654	3572	992	992	73
283*	1530	3364	1024	1056	73
284	3261	2751	1024	1024	74
285*	2543	1722	1056	1056	74

286*	1306	3645	1024	1056	74
287*	2614	3512	1056	1056	74
288*	1430	3224	1024	1056	74
289	3061	2451	1024	1024	75
290	2143	1122	1024	1024	76
291*	0307	2244	992	992	76
292*	0616	0511	1024	992	76
293*	1435	1223	992	992	76
294*	3073	2446	992	1024	76
295*	2167	1114	1024	1056	76
296	0357	2230	1024	1024	77
297	0736	0461	1024	1024	78
298*	1675	1143	1056	1056	78
299*	3573	2306	1024	992	78
300*	3366	0615	1056	1056	78
301*	2755	1433	1024	992	78
302	1732	3067	1024	1024	79
303*	3665	2157	1056	1056	79
304*	3552	0337	1056	1024	79
305*	3324	0676	1056	1024	79
306*	2651	1575	1024	992	79
307*	1522	3373	1056	992	79
308*	3245	2767	1024	992	79
309	2513	1756	1024	1024	80
310	1226	3735	1024	1024	81
311*	2454	3672	1056	1056	81
312*	1130	3564	992	1024	81
313*	2260	3350	1056	1024	81
314*	0541	2721	1024	1056	81
315*	1303	1642	992	1056	81
316*	2606	3505	1024	1056	81
317	1414	3212	1024	1024	82
318*	3031	2425	992	992	82
319*	2063	1052	1024	992	82
320*	0147	2124	1056	1056	82
321*	0316	0251	1056	1024	82
322*	0634	0522	1024	992	82
323*	1471	1245	992	992	82
324*	3163	2512	1024	1056	82
325	2347	1224	1024	1024	83
326*	0717	2450	1056	992	83
327*	1637	1120	1056	1024	83

328*	3477	2240	1024	1056	83
329	3176	0501	1024	1024	84
330*	2375	1203	992	1056	84
331*	0773	2406	992	1024	84
332*	1767	1014	1024	992	84
333	3757	2030	1024	1024	85
334*	3736	0061	1056	992	85
335*	3674	0142	1056	1024	85
336*	3570	0304	992	1024	85
337*	3360	0610	1056	1024	85
338*	2741	1421	992	1024	85
339*	1702	3043	1056	1024	85
340*	3605	2107	992	1024	85
341*	3412	0217	1056	1024	85
342*	3024	0436	1056	1024	85
343*	2051	1075	992	1024	85
344*	0123	2172	992	1024	85
345*	0246	0365	1024	1056	85
346	0514	0752	1024	1024	86
347	1231	1725	1024	1024	87
348	2462	3653	1024	1024	88
349	1144	3526	1024	1024	89
350	2310	3254	1024	1024	90
351	0621	2531	1024	1024	91
352	1443	1262	1024	1024	92
353	3107	2544	1024	1024	93
354*	2217	1310	1056	1056	93
355*	0437	2620	992	1024	93
356*	1077	1440	1024	992	93
357	2176	3101	1024	1024	94
358	0375	2203	1024	1024	95
359	0772	0407	1024	1024	96
360	1765	1017	1024	1024	97
361	3753	2036	1024	1024	98
362	3726	0075	1024	1024	99
363*	3654	0172	992	1056	99
364*	3530	0364	1056	1024	99
365*	3260	0750	992	1024	99
366*	2541	1721	1056	1024	99
367*	1302	3643	1024	992	99
368	2604	3506	1024	1024	100
369	1410	3214	1024	1024	101

370	3021	2431	1024	1024	102
371	2043	1062	1024	1024	103
372*	0107	2144	992	992	103
373*	0216	0311	1056	1024	103
374*	0434	0622	1024	1056	103
375	1071	1445	1024	1024	104
376*	2162	3113	992	992	104
377*	0345	2227	992	1024	104
378*	0712	0457	1024	1056	104
379	1625	1137	1024	1024	105
380	3453	2276	1024	1024	106
381*	3126	0575	1056	1056	106
382*	2255	1373	1056	1024	106
383*	0533	2766	992	1024	106
384*	1267	1754	1056	1024	106
385*	2556	3731	992	1024	106
386*	1334	3662	992	1024	106
387*	2670	3544	1056	1024	106
388*	1560	3310	1056	1024	106
389*	3341	2621	1024	992	106
390*	2703	1442	1056	992	106
391*	1606	3105	1056	1024	106
392*	3415	2213	1024	1056	106
393	3032	0427	1024	1024	107
394	2065	1057	1024	1024	108
395	0153	2136	1024	1024	109
396	0326	0275	1024	1024	110
397	0654	0572	1024	1024	111
398*	1531	1365	992	992	111
399*	3263	2752	1024	992	111
400*	2547	1724	992	992	111
401*	1316	3651	992	1024	111
402*	2634	3522	992	1024	111
403*	1470	3244	1024	1056	111
404	3161	2511	1024	1024	112
405	2343	1222	1024	1024	113
406	0707	2444	1024	1024	114
407*	1617	1110	992	992	114
408*	3437	2220	1024	1056	114
409	3076	0441	1024	1024	115
410*	2175	1103	992	1056	115
411*	0373	2206	1024	1056	115

412*	0766	0415	1056	992	115
413*	1755	1033	1056	1024	115
414*	3733	2066	1024	1056	115
415	3666	0155	1024	1024	116
416*	3554	0332	1056	992	116
417*	3330	0664	1024	1056	116
418*	2661	1551	1056	992	116
419*	1542	3323	992	1024	116
420*	3305	2647	1024	1056	116
421	2613	1516	1024	1024	117
422*	1426	3235	1056	1056	117
423*	3055	2473	1056	1024	117
424*	2133	1166	1056	1024	117
425*	0267	2354	992	1024	117
426*	0556	0731	1024	1056	117
427	1335	1663	1024	1024	118
428*	2672	3547	992	1056	118
429*	1564	3316	1056	1024	118
430*	3351	2635	1056	1024	118
431*	2723	1472	1056	1024	118
432*	1646	3165	1056	1024	118
433*	3515	2353	1024	992	118
434	3232	0727	1024	1024	119
435*	2465	1657	1056	992	119
436*	1152	3537	992	1024	119
437*	2324	3276	992	1024	119
438*	0651	2575	992	1024	119
439*	1523	1372	1024	1056	119
440	3247	2764	1024	1024	120
441	2517	1750	1024	1024	121
442*	1236	3721	1056	992	121
443*	2474	3642	1056	1024	121
444*	1170	3504	1056	1024	121
445*	2360	3210	1056	1024	121
446*	0741	2421	1024	992	121
447	1703	1042	1024	1024	122
448*	3607	2104	1056	992	122
449*	3416	0211	992	1024	122
450*	3034	0422	1024	992	122
451*	2071	1045	1056	992	122
452*	0163	2112	1056	1024	122
453*	0346	0225	1024	992	122

454	0714	0452	1024	1024	123
455*	1631	1125	1056	992	123
456*	3463	2252	992	1024	123
457*	3146	0525	1056	1024	123
458*	2315	1253	992	1024	123
459*	0633	2526	1056	1024	123
460*	1467	1254	1024	992	123
461*	3157	2530	1056	1056	123
462*	2337	1260	992	1024	123
463*	0677	2540	992	1024	123
464*	1577	1300	1056	1024	123
465*	3377	2600	992	1024	123
466*	2777	1400	1024	992	123

4.0 SPECTRAL PROPERTIES OF FORWARD LINK COMMAND CHANNEL AND MODE 2 RETURN LINK CODES

4.1 Introduction

One of the reasons for encoding transmissions in the TDRSS is to achieve signals which satisfy flux density restrictions imposed by the International Radio Advisory Committee (IRAC). The maximal linear sequences such as were selected for the encoding of the Mode 1 Return Link are ideal in this respect since the spectral lines generated by such codes are of uniform amplitude over the entire frequency band (except at multiples of the code repetition frequency) and hence the spectrum of the resultant transmissions consist of spectral lines which follow a $\sin^2 x/x^2$ envelope.

Since the pseudo noise codes which were selected for the forward link command channel and the Mode 2 return link are Gold codes, they are not maximal. The spectral lines generated by these codes are thus not of uniform amplitude and hence it is of interest to compare spectral lines of the signal resulting from the use of these codes with those obtained using maximal codes of the same length.

4.1 Description of Results

The spectra of the forward link command channel and Mode 2 return link codes were computed and, for each case, the maximum spectral line was compared with the maximal spectral line for the case of a maximal pseudo noise code of the same length. The ratio,

$$10 \log \frac{1}{p+1} \max_{j} \left[\frac{|F(a)|^2 (j) \sin^2 (\frac{\pi j}{p})}{(\frac{\pi j}{p})^2} \right] ,$$

where a is the code and p is the code period, is a measure of the relative performance of the nonmaximal code libraries in achieving low flux density when compared with the maximal PN codes.

Table contains the above ratio (listed under the columns labeled "Spectral Increase DB") for each user code for the forward link command channel and the Mode 2 return link I and Q channels.

The computer program used to generate the forward link command channel and Mode 2 return link codes and to compute the above ratio via the Fast Fourier Transform is included in Appendix A.

USER CODE ASSIGNMENT	FORWARD LINK COMMAND CHANNEL	SPECTRAL INCREASE DB	MODE 2 RETURN LINK I CHANNEL	SPECTRAL INCREASE DB	MODE 2 RETURN LINK Q CHANNEL	SPECTRAL INCREASE DB
1	1010011	7.3	0004	7.3	0006	7.3
2	0110011	7.7	0042	7.7	0262	7.8
3	1100000	7.6	0100	7.3	0140	8.5
4	0010000	7.3	0200	7.5	0300	7.8
5	1010000	7.0	0240	8.5	0360	8.2
6	0110000	7.3	1004	7.5	3400	8.2
7	1110000	6.2	2010	7.9	3014	7.3
8	0001000	7.6	0104	7.9	0146	6.9
9	1001000	6.2	2522	8.3	3773	7.7
10	0101000	7.7	1244	7.4	3766	6.9
11	1001011	7.5	2440	7.2	3660	7.2
12	0011000	7.0	1100	7.9	3540	6.6
13	0101011	7.3	3203	8.1	2702	8.3
14	0011011	7.9	0071	8.0	2045	8.4
15	1111000	7.8	0162	8.3	0113	7.7
16	1011011	8.2	0344	7.6	0226	7.8
17	0111011	7.5	0710	7.1	045[illegible]	8.1
18	0100100	8.0	1621	7.7	1131	8.1
19	1111011	7.2	0335	7.4	2263	8.3
20	0000111	7.5	0672	8.2	0547	8.5
21	1010100	7.7	1565	8.3	1317	8.4
22	1000111	7.7	3535	8.1	2363	7.4
23	0100111	8.4	3272	7.4	0747	7.5
24	0001100	7.5	1650	7.6	3174	6.9
25	1001100	7.5	2052	7.5	3074	7.8
26	0101100	7.0	2422	8.4	3633	8.1
27	0010111	8.1	1044	7.6	3406	7.6
28	0011100	7.3	2110	7.7	3154	7.4
29	1011100	8.6	0425	7.9	2637	8.4

Table 4. Spectral Increase Due to Nonmaximal Codes

Table 4. (continued)

USER CODE ASSIGNMENT	FORWARD LINK COMMAND CHANNEL	SPECTRAL INCREASE DB	MODE 2 RETURN LINK I CHANNEL	SPECTRAL INCREASE DB	MODE 2 RETURN LINK Q CHANNEL	SPECTRAL INCREASE DB
30	0111100	6.7	1053	9.0	1476	7.4
31	0110111	8.5	0532	7.8	0767	8.0
32	0000010	7.6	2047	7.5	1064	8.0
33	1000010	7.2	0117	8.4	2150	8.0
34	0100010	7.1	1171	7.7	1505	7.0
35	1100010	7.4	3471	6.9	2245	7.7
36	1110111	6.2	3457	7.0	2270	7.2
37	1010010	7.6	3136	8.2	0561	9.1
38	0001111	7.5	3671	7.1	2145	7.6
39	1110010	7.1	3562	7.5	0313	7.3
40	0001010	8.2	2225	8.4	1337	8.4
41	1001010	8.5	2566	7.7	3715	8.1
42	0101111	8.4	3025	7.8	2437	8.4
43	1101010	7.6	1271	7.7	1745	7.8
44	0011010	8.6	2562	7.9	3713	6.9
45	1101111	7.4	1344	7.6	3626	7.5
46	0011111	6.9	0135	7.0	2163	7.1
47	1011111	8.0	2722	7.5	3473	7.7
48	1111111	7.4	2466	8.1	3655	8.5
49	1000110	6.2	0661	7.9	2551	7.1
50	0110101	7.1	1543	8.3	1322	8.1
51	1110101	7.3	3307	8.2	2644	8.2
52	0010110	7.3	2617	7.4	1510	7.0
53	0001101	6.6	1436	8.3	3221	8.1
54	0110110	7.1	3566	7.9	0315	7.5
55	1101101	9.0	3354	7.6	2632	8.5
56	0001110	6.6	2731	7.6	1465	7.3
57	0011101	6.2	3125	8.3	2577	6.6

Table 4. (continued)

USER CODE ASSIGNMENT	FORWARD LINK COMMAND CHANNEL	SPECTRAL INCREASE DB	MODE 2 RETURN LINK I CHANNEL	SPECTRAL INCREASE DB	MODE 2 RETURN LINK Q CHANNEL	SPECTRAL INCREASE DB
58	0101110	7.7	1257	7.6	1770	7.2
59	1101110	6.7	2740	7.3	3420	7.9
60	1011101	6.5	1371	8.3	1605	7.6
61	1011110	7.3	2762	8.4	3413	7.5
62	0111110	6.9	3622	6.9	0133	7.1
63	1111110	6.9	3444	8.3	0266	7.6
64	0000001	8.8	3110	7.8	0554	7.8
65	1000001	8.2	2221	8.2	1331	7.8
66	1111101	8.5	0443	7.4	2062	7.7
67	1100001	6.2	0732	7.3	0467	7.5
68	0010001	7.6	1665	7.2	1157	8.2
69	0000011	8.6	3553	7.6	2336	7.2
70	0110001	7.2	1532	8.3	3367	8.2
71	1110001	7.1	3265	7.4	2757	7.4
72	0001001	9.6	2553	7.8	1736	7.7
73	1000011	8.9	1326	8.5	3675	6.9
74	0100011	7.8	3261	8.1	2751	8.4
75	1101001	8.4	3061	7.6	2451	7.6
76	0011001	7.7	2143	7.9	1122	7.5
77	1011001	7.5	0357	7.9	2230	7.8
78	0111001	6.8	0736	7.4	0461	7.7
79	1111001	7.1	1732	7.7	3067	7.2
80	0000101	7.3	2513	7.6	1756	7.2
81	1000101	7.4	1226	7.7	3735	6.8
82	1100011	8.3	1414	8.7	3212	7.2
83	1100101	9.1	2347	7.2	1224	7.7
84	0010011	6.5	3176	8.3	0531	8.3
85	1010101	6.0	3757	7.4	2030	8.4

REFERENCE

1. TDRSS Telecommunications Study: Phase I - Final Report, Report No. R-4958, The Magnavox Company, 15 September 1974.

APPENDIX A

COMPUTER ROUTINES FOR COMPUTATION OF CODE PERFORMANCE

```
UNX COMPILER (VER.2.3M)                    03/16/77. 20.24.41.

                PROGRAM RGA27(TAPE5,TAPE55,TAPE6)
           *
           *  SPECTRAL INCREASE OF TDRSS PN CODE LIBRARY
           *
00004           DIMENSION IN(100,3),SPECIN(3)
           *
           *  READ IN DATA
           *
           *  BEGINNING USER CODE FOLLOWED BY THE TRIPLET SETS OF
           *    INITIAL CONDITIONS
           *
00004           REWIND 5
00006           ACCEPT(5) IUSER
00013           I = 0
00014     105   CONTINUE
00014           I = I + 1
00026           ACCEPT(5) (IN(I,J),J=1,3)
00031           IF (EOF(5)) 110,105
00035     110   CONTINUE
00035           INUM = I - 1
00037           WRITE(6,120)
00043     120   FORMAT(///// 14X *FORWARD* 17X *MODE 2* 17X *MODE 2*/
00043         1        16X *LINK* 4X *SPECTRAL* 6X *RETURN*
00043         2        3X *SPECTRAL* 6X *RETURN* 3X *SPECTRAL*/
00043         3     * USER CODE* 4X *COMMAND* 3X *INCREASE*
00043         4     6X *LINK I* 3X *INCREASE* 6X *LINK Q* 3X *INCREASE*/
00043         5     *ASSIGNMENT* 4X *CHANNEL* 6X *DB* 8X *CHANNEL*
00043         6     6X *DB* 8X *CHANNEL* 6X *DB*//)
00043           DO 200 I=1,INUM
00043           CALL CNVBIN(IN(I,1),IHA)
00047           IHA = 2*IHA
00050           CALL SSDR(IHA,3515B,10,1023,1110B,2011B,10,1023,1023)
00061           CALL FFTDR(7HCOMPLEX,-1,1023,0,0,SPECIN(1))
00065           CALL SSDR(IN(I,2),4005B,11,2047,1B,4445B,11,2047,2047)
00076           CALL FFTDR(7HCOMPLEX,-1,2047,0,0,SPECIN(2))
00102           CALL SSDR(IN(I,3),4005B,11,2047,1B,4445B,11,2047,2047)
00113           CALL FFTDR(7HCOMPLEX,-1,2047,0,0,SPECIN(3))
```

```
UNX COMPILER (VER.2.3M)                    03/16/77. 20.24.41.          RGA27

00117           CALL IFILIN(7,IN(I,1),IN(I,2),IERROR)
00124           WRITE(6,150) IUSER,(IN(I,J),SPECIN(J),J=1,3)
00144     150   FORMAT(4X I2, 3X A7, F8.1, 10X O4, F9.1, 10X O4,
00144          1         F9.1)
00144           IUSER = IUSER + 1
00146     200   CONTINUE
00150           STOP
00152           END
```

```
UNX COMPILER (VER.2.3M)                    03/16/77. 20.24.41.

            SUBROUTINE CNVBIN(IN,IH)
       *
       *  IN IS THE INTEGER REPRESENTATION OF THE BINARY NUMBER
       *
       *  IH IS THE RESULTING INTEGER
       *
            DIMENSION INUM(14)
       *
       *  SPLIT OUT TERMS FROM THE NUMBER
       *
            INT = IN
            DO 105 I=1,14
            ITEMP = INT
            INT = INT/10
            INUM(I) = ITEMP - 10*INT
      105   CONTINUE
       *
       *  MULTIPLY EACH TERM BY THE APPROPRIATE POWER
       *
            IH = 0
            DO 110 I=1,14
            IH = 2*(IH+INUM(15-I))
      110   CONTINUE
            IH = IH/2
            RETURN
            END
```

```
:UNX COMPILER (VER.2.3M)                    03/16/77. 20.24.41.

              SUBROUTINE SSDR(IHA,IFA,NA,IPERA,IHB,IFB,NB,IPERB,IPERIOD)
        *
        *
        *  SUM SEQUENCE DATA GENERATOR SUBROUTINE.
        *
        *
        *  ALL INPUT PARAMETERS MUST BE SPECIFIED.
        *    NOTE: PERIOD INPUTS--0 IMPLIES MAXIMAL SEQUENCES.
        *
        *  SUM CODE OF TWO SHIFT REGISTERS, NOT NECESSARILY OF EQUAL LENGTH,
        *    IE. G1/F1 + G2/F2  =  (G1*F2 + G2*F1)  /  F1*F2
        *
        *  AUTHORS--IRA GREEN AND ROBERT GOLD
        *
        *  THE SUM OF THE STAGES OF A AND B SHIFT REGISTERS MUST
        *    NOT EXCEED 59.
        *
        *    H IS INITIAL CONDITION SHIFT REGISTER CONFIGURATION
        *    G IS INITIAL CONDITION POLYNOMIAL
        *    F IS ASSOCIATED POLYNOMIAL
        *
        *
        *
)00014          COMMON /SSDATA/ IH1(60),IG1(60),IF1(60),
)00014         1         IH2(60),IG2(60),IF2(60),
)00014         2         IF12(120),IG(120),ITP1(120),ITP2(120),
)00014         3         IDATA(2047)
        *
        *  IHA, IFA, IHB AND IFB ARE TO BE INPUT IN OCTAL.
        *
        *
        *
        *  NA AND NB ARE THE ORDERS OF THE A AND B SHIFT REGISTERS.
        *
        *  IPERA IS THE PERIOD OF THE A SHIFT REGISTER.
        *
        *  IPERB IS THE PERIOD OF THE B SHIFT REGISTER.
```

```
            *
            *  IPERIOD IS THE PERIOD OF THE SUM SEQUENCE FROM THE A AND B
            *    SHIFT REGISTERS.
            *
            *
            *  PREPARE TO STORE SUM SEQUENCE
            *
000014            REWIND 55
000016            CALL FTNBIN(0,1,55)
000021            IF (IPERA.EQ.0) IPERA = 2**NA-1
000032            IF (IPERB.EQ.0) IPERB = 2**NB-1
000041            IF (IPERIOD.EQ.0)  IPERIOD = IPERA * IPERB
            *
            *  CALCULATE NUMBER OF TERMS FROM THE ORDER
            *
000045            N1 = NA + 1
000046            N2 = NB + 1
            *
            *  CALCULATE NUMBER OF TERMS IN THE PRODUCT POLYNOMIAL-- THE
            *    GENERATOR OF THE SUM SEQUENCE.
            *
000050            NT = NA + NB + 1
            *
            *  INPUT OCTAL REPRESENTATION IS CONVERTED TO A BINARY COEFFICIENT
            *    TABLE WITH THE FIRST ELEMENT BEING THE CONSTANT TERM.
            *      1ST ARG = OCTAL REPRESENTATION
            *      2ND ARG = OUTPUT TABLE OF BINARY COEFFICIENTS
            *      3RD ARG = NUMBER OF TERMS
            *
000052            CALL BINCOEF(IHA,IH1,N1)
000054            CALL BINCOEF(IFA,IF1,N1)
000062            CALL BINCOEF(IHB,IH2,N2)
000070            CALL BINCOEF(IFB,IF2,N2)
000076            CALL POLYBX(IF1,IF2,IF12,N1,N2)
            *  ZERO TERM ADDED AT THE END OF IF12
            *    MAKING THE NUMBER OF TERMS = N1 + N2
            *
            *  IF12 MUST HAVE A CONSTANT TERM
```

```
        *
        *
        *  COMPACT IF12 POLYNOMIAL INTO ONE WORD
000102        CALL CPACT(IF12,IFF,NT)
        *  IFF MUST HAVE A CONSTANT TERM
        *    IFF = F1 * F2
        *
        *
        *  INITIAL CONDITION POLYNOMIAL IS OBTAINED FROM INITIAL CONDITIONS
        *    FOR SHIFT REGISTER A AND THEN MULTIPLIED BY THE ASSOCIATED
        *    POLYNOMIAL FOR B TO OBTAIN IG1*IF2.
        *
000105        CALL INITIAL(IG1,IF1,IH1,N1)
000110        CALL POLYBX(IG1,IF2,ITP1,N1,N2)
        *
        *  INITIAL CONDITION POLYNOMIAL IS OBTAINED FROM INITIAL CONDITIONS
        *    FOR SHIFT REGISTER B AND THEN MULTIPLIED BY THE ASSOCIATED
        *    POLYNOMIAL FOR A TO OBTAIN IG2*IF1.
        *
000114        CALL INITIAL(IG2,IF2,IH2,N2)
000117        CALL POLYBX(IG2,IF1,ITP2,N2,N1)
        *
        *  ADD THE POLYNOMIALS
        *
000123        CALL PADD(ITP1,ITP2,IG,NT)
        *  ZERO TERM ADDED AT THE END OF IG
        *    MAKING THE NUMBER OF TERMS = NT + 1
        *
        *
000126        CALL CPACT(IG,IGG,NT)
        *
        *  IGG = IG1*IF2 + IG2*IF1
        *
000131        ILEFT = IPERIOD
000133    110 CONTINUE
000133        IF (ILEFT.LE.2047) GO TO 150
000141        ILEFT = ILEFT - 2047
000142        CALL SSIDATA(IGG,IFF,2047,IDATA)
```

```
RUNX COMPILER (VER.2.3M)                03/16/77. 20.24.41.        SSDR

000145          WRITE(55) IDATA
000152          GO TO 110
000156      150 CONTINUE
000156          CALL SSIDATA(IGG,IFF,ILEFT,IDATA)
000161          WRITE(55) (IDATA(I),I=1,ILEFT)
000173          RETURN
000174          END
```

```
RUNX COMPILER (VER.2.3M)                03/16/77. 20.24.41.

                 SUBROUTINE CPACT(IP,IPC,NN)
         *
         *  COMPACT IP POLYNOMIAL WITH NN TERMS INTO ONE WORD IPC
         *
         *    SIGN BIT IS LOW ORDER TERM OF POLYNOMIAL
         *
000006           DIMENSION IP(1)
000006           IPC = 0
000006           DO 100 I=1,NN
000010           CALL SBYT(61-I,1,IPC,IP(I))
000017   100     CONTINUE
000024           RETURN
000024           END
```

```
RUNX COMPILER (VER.2.3M)                    03/16/77. 20.24.41.

                SUBROUTINE PADD(IA,IB,IC,NN)
            *
            *  IC = IA (+) IB WITH ZERO TERM ADDED AT THE END MAKING
            *                 THE NUMBER OF TERMS = NN+1
            *
000007          DIMENSION IA(1),IB(1),IC(1)
000007          DO 100 I=1,NN
000010          IC(I) = IA(I) + IB(I) -IA(I)*IB(I)*2
000017   100    CONTINUE
000022          IC(NN+1) = 0
000023          RETURN
000023          END
```

```
RUNX COMPILER (VER.2.3M)                    03/16/77. 20.24.41.

                SUBROUTINE INITIAL(IG,IF,IH,NN)
          *
          *  CALCULATE INITIAL CONDITION POLYNOMIAL IG FROM
          *     INITIAL CONDITIONS OF SHIFT REGISTER IH AND
          *     ASSOCIATED POLYNOMIAL IF
          *  NN IS THE ORDER+1
          *
000007          DIMENSION IG(1),IF(1),IH(1)
000007          DO 200 I=1,NN
000010          IG(I) = 0
000011    200   CONTINUE
000013          NNM1 = NN - 1
000015          DO 500 IO=1,NNM1
000015          DO 400 II=1,IO
000016          IJ = IO - II + 1
000021          IT = IH(II) * IF(IJ)
000025          IG(IO) = IG(IO) + IT - IG(IO)*IT*2
000032    400   CONTINUE
000035    500   CONTINUE
000037          RETURN
000040          END
```

```
RUNX COMPILER (VER.2.3M)                    03/16/77. 20.24.41.

                SUBROUTINE   POLYBX(IA,IB,IC,NA,NB)
000010          DIMENSION IA(1),IB(1),IC(1)
         *
         *
         *  NA AND NB ARE THE NUMBER OF TERMS IN THE A AND B
         *    POLYNOMIAL RESPECTIVELY.  THE RESULTS WILL HAVE NA + NB
         *    TERMS, WHICH INCLUDES A ZERO TERM ADDED AT THE END.
         *
         *  ZERO OUT RESULT
         *
         *
000010          NAB = NA + NB
000011          DO 10 I = 1,NAB
000011          IC(I) = 0
000014   10     CONTINUE
         *
000017          DO 30 I=1,NA
         *
000020          IAI = IA(I)
000021          IF(IAI.EQ.0)   GO TO 30
000023          I1 = I - 1
         *
         *
         *  MULTIPLY BY THE ITH TERM OF THE A POLYNOMIAL
         *
000025          DO 20 J= 1,NB
         *
         *
         *  KTH TERM BEING CALCULATED
         *
000026          K = I1 + J
000027          IC(K) = IC(K) + IB(J)
000033   20     CONTINUE
         *
000035   30     CONTINUE
         *
         *
```

```
RUNX COMPILER (VER.2.3M)                03/16/77. 20.24.41.          POLYBX

               *   TAKE RESULT MOD 2
               *
000040             DO 40 I = 1,NAB
000040             ICI = IC(I)
000042             ICH = ICI/2
000044             IC(I) = ICI - 2*ICH
000047       40    CONTINUE
               *
000051             RETURN
000052             END
```

```
RUNX COMPILER (VER.2.3M                    03/16/77. 20.24.41.

                  SUBROUTINE BINCOEF(IP,IPA,NA)
000006            DIMENSION IPA(1)
           *
000006            IH = IP
           *
           *
000007            DO 10 I=1,NA
000010            IPA(I) = IH
000012            IH = IH/2
000013    10      CONTINUE
           *
000015            DO 20 I=1,NA
000016            I7 = IPA(I)
000017            IH = IZ/2
000021            IPA(I) = IZ - 2*IH
000024    20      CONTINUE
           *
000026            RETURN
000026            END.
```

```
STORAGE ALLOCATION.

     ADDRESS   LENGTH          BINARY CONTROL CARDS.

           0        10          IDENT   SSIDATA
          10                    END

                      ENTRY POINTS.

                      SSIDATA   -      1
```

```
                                          IDENT  SSIDATA
                                          ENTRY  SSIDATA
                                 ***
                                 *  ARGUMENTS ARE:
                                 *    C(B1) G = INITIAL CONDITIONS POLYNOMIAL-ALTERED+-
                                 *              IF SSIDATA IS CALLED AGAIN, THE SEQUENCE WILL
                                 *              CONTINUE AT TERM N + 1.
                                 *    C(B2) F = ASSOCIATED POLYNOMIAL
                                 *    C(B3) N = NUMBER OF TERMS OF THE SEQUENCE TO BE STORED IN IDATA
                                 *    C(B4) IDATA = FIRST TERM OF GENERATED SEQUENCE--A SERIES
                                 *                  OF 0*S AND 1*S
                                 *
                                 *    B5 = COUNTER FOR ELEMENT OF ARRAY IDATA
                                 *    B6 = N
                                 *    B7 = 1
                                 *
                                 *    X1 = G
                                 *    X2 = F
                                 *    X3 = N
                                 ***
0 23230705160000000004                    VFD    42/0LSSGEN,18/4
1                              SSIDATA    BSS    1            ENTRY/EXIT LINE
2 6170000001                              SB7    1            PUT 1 IN B7
           56110                          SA1    B1           FETCH G
                56220                     SA2    B2           FETCH F
3 56330                                   SA3    B3           FETCH N TO X3
      63630                               SB6    X3            AND PUT IN B6
           65500                          SB5    B0           INITIALIZE ARRAY COUNTER
                43600                     MX6    0            PUT 0 IN X6
4 0321000005+                  LOOP       PL     X1,NONE      IF LOW ORDER OF G IS 0 SHIFT ONLY
           13112                          BX1    X1-X2        OTHERWISE DIVIDE
                                 *
                                 *                            CONSTANT TERM GUARANTEED IN
                                 *                            ASSOCIATED POLYNOMIAL
                                 *                            REMOVES LOW ORDER
                                 *
                76670                     SX6    B7           PREPARE TO STORE A 1
```

57

```
    5 20101                    NONE      LX1    1              NOW SHIFT
           56645                         SA6    B4+B5          STORE TERM
                43600                    MX6    0              RESET X6 TO 0
                     66557               SB5    B5+B7          INCREMENT LOOP COUNT
    6 0756000004+                        LT     B5,B6,LOOP     IF COUNTER LESS THAN N,LOOP
                10711                    BX7    X1             OTHERWISE, STORE LATEST
                     56710               SA7    B1              INITIAL CONDITIONS POLYNOMIAL AND
    7 0400000001+                        EQ     SSIDATA        RETURN
   10                                    END

          43155 STORAGE USED        46 STATEMENTS        3 SYMBOLS
```

```
              SUBROUTINE FFTDR(ITYPE,ISIGN,ILENGTH,IFTADJ,IOTERMS,SPECIN)
          *
          *  GENERAL FAST FOURIER TRANSFORM AND ITS INVERSE OF SEQUENCE
          *
          *  ALL INPUT PARAMETERS MUST BE SPECIFIED.
          *
          *  AUTHORS--IRA GREEN AND ROBERT GOLD
          *
          *    MAXIMUM LENGTH OF SEQUENCE IS 4998 REAL OR 2500 COMPLEX
          *
          *
          *    4998/2 + 1 = 2500
000011          COMPLEX TRAN(2500)
          *
          *    2500 * 2 = 5000
000011          COMMON /FFTDATA/ DATA(5000)
000011          DIMENSION IDATA(5000)
000011          EQUIVALENCE (DATA,IDATA,TRAN)
000011          PI = 4.0 * ATAN(1.0)
          *
          *
          *
          *  ITYPE IS THE TYPE OF CACULATION, ≠REAL≠ OR ≠COMPLEX≠
          *
          *  ISIGN IS THE SIGN OF THE TRANSFORM EXPONENTIAL, -1 OR +1
          *
          *  ILENGTH IS THE NUMBER OF TERMS IN THE SEQUENCE OR ITS PERIOD ≠P≠
          *
          *  IFTADJ IS THE NUMBER OF TERMS THE SEQUENCE MUST BE ADJUSTED
          *    BY FOR FAST FOURIER TRANSFORM REQUIREMENTS.  TERMS ARE ADDED
          *    BY CLOCKING THE SEQUENCE PAST THE PERIODIC POINT, WHEN IFTADJ
          *    IS POSITIVE.
          *
          *  IOTERMS IS THE NUMBER OF TERMS OF THE FAST FOURIER
          *    TRANSFORM RESULT TO BE DISPLAYED
          *
          *  INPUT PARAMETER LIST CONTAINS ITYPE, ISIGN, ILENGTH, IFTADJ, AND
```

```
        *    IOTERMS AND TAPE55 CONTAINS THE INPUT SEQUENCE IN UNFORMATTED
        *    BINARY
        *
        *   ILENGTH+IOTERMS MUST BE A POSITIVE EVEN INTEGER,NONE
        *    OF WHOSE PRIME FACTORS IS GREATER THAN 97.  THERE
        *    IS A MAXIMUM OF 32 PRIME FACTORS, ONLY 10 OF WHICH
        *    MAY BE UNPAIRED, WHEN THE SEQUENCE IS REAL.  FOR COMPLEX
        *    SEQUENCES THE ≠EVEN≠ RESTRICTION IS REMOVED.
        *
        *
000014        CALL SECOND(T1)
        *
        *
        *
        *   PREPARE TO READ SUM SEQUENCE
        *
000015        REWIND 55
        *
        *
        *   GENERATE SEQUENCE IN MEMORY AS DATA
        *    THE SEQUENCE IS ADJUSTED BY CLOCKING TO
        *    MEET REQUIREMENTS OF THE FAST FOURIER TRANSFORM
        *    PROGRAM.  IN ADDITION 0≠S ARE CONVERTED TO +1
        *    AND 1≠S ARE CONVERTED TO -1 WHICH WILL YIELD
        *    THETA AS THE CONSTANT TERM OF THE FAST FOURIER
        *    TRANSFORM.  NOTE THAT ITS VALUE IS
        *    ALTERED DUE TO THE AUGMENTED SUM SEQUENCE.
        *
000017        READ(55) (IDATA(I),I=1,ILENGTH)
        *
        *     ACCEPT(55) (DATA(I),I=1,ILENGTH)
        *     GO TO 200
        *
        *
        *   THE ENTIRE SEQUENCE IS NEEDED FOR THE EVALUATION
        *    OF THETA.
        *
000041        DO 120 I=1,ILENGTH
```

```
RUNX COMPILER (VER.2.3M)                    03/16/77. 20.24.42.          FFTOR

000046              IF (IDATA(I).EQ.0) GO TO 110
000047              DATA(I) = -1.0
000051              GO TO 120
000051      110     CONTINUE
000051              DATA(I) = +1.0
000053      120     CONTINUE
            *
            *  IF NECESSARY, CLOCK SEQUENCE TO AUGUMENT ITS SIZE
            *
000056              IF (IFTADJ.LE.0) GO TO 200
000057              DO 140 I=1,IFTADJ
000061              DATA(ILENGTH+I) = DATA(I)
000064      140     CONTINUE
000066      200     CONTINUE
000066              IM2 = MINO(100,ILENGTH)
000071              DO 210 I=1,IM2
000071              IDATA(I) = DATA(I)
000076      210     CONTINUE
000100              IF (IOTFRMS.EQ.0) GO TO 305
000101              WRITE(6,300) (IDATA(I),I=1,IM2)
000113      300     FORMAT(//* SEQUENCE UP TO FIRST ONE HUNDRED TERMS*/
000113      +       (25I2) )
000113       305    CONTINUE
000113              DO 310 I=1,IM2
000117              DATA(I) = IDATA(I)
000122      310     CONTINUE
            *
            *  EVALUATE FIRST TERM OF FAST FOURIER TRANSFORM OR THETA
            *     AND ADJUSTED THETA DUE TO AUGMENTED SIZE OF SEQUENCE
            *
000125              THETA = 0.0
000126              DO 320 I=1,ILENGTH
000127              THETA = THETA + DATA(I)
000131      320     CONTINUE
000133              THETAA = THETA
000134              IF (IFTADJ) 330,350,340
000135      330     M2 = -IFTADJ
000136              DO 335 I=1,M2
```

```
RUNX COMPILER (VER.2.3M)                    03/15/77. 20.24.42.          FFTDR

000136          THETAA = THETAA - DATA(ILENGTH-I+1)
000143   335    CONTINUE
000145          GO TO 350
000146   340    CONTINUE
000146          DO 345 I=1,IFTADJ
000150          THETAA = THETAA + DATA(ILENGTH+I)
000153   345    CONTINUE
000155   350    CONTINUE
000155          ITHETA = THETA
000157          ITHETAA = THETAA
000161          IF (IOTERMS.EQ.0) GO TO 370
000162          WRITE(6,360) ITHETA,ITHETAA
000172   360    FORMAT(//* THETA AND THETA OF AUGMENTED SEQUENCE IS*
000172   +      I5 *  AND* I5)
000172   370    CONTINUE
000172          IF (ITYPE.EQ.7HCOMPLEX) GO TO 400
000200          IFORM = 0
000201          GO TO 500
000201   400    CONTINUE
000201          IFORM = 1
000202          IM1 = 0
000203          IM2 = ILENGTH - 1
000205          DO 420 I=IM1,IM2
000205          J = ILENGTH - I
000210          TRAN(J) = DATA(J)
000213   420    CONTINUE
000216   500    CALL FOURT(DATA,ILENGTH+IFTADJ,1,ISIGN,IFORM)
         *
         *  COMPLEX FAST FOURIER TRANSFORM
         *
000224          IF (IOTERMS.EQ.0) GO TO 570
000231          WRITE(6,550) (TRAN(I),I=1,IOTERMS)
000251   550    FORMAT(//* COMPLEX FAST FOURIER TRANSFORM OF SEQUENCE*/
000251        1         ( 5( *(* F6.1 *,* F6.1 *)*) ) )
000251   570    CONTINUE
         *
         *  OBTAIN SQUARE OF ABSOLUTE VALUE OF FAST FOURIER TRANSFORM
         *
```

```
RUNX COMPILER (VER.2.3M)                 03/16/77. 20.24.42.          FFTDR

000251          IO2 = 2 * IOTERMS
000256          IL2 = 2*ILENGTH
000257          IF (IFORM.NE.1) IL2 = ILENGTH + 2
000263          IM2 = MINO(IO2,IL2)
000267          DO 600 I=1,IL2,2
000271          DATA(I) = DATA(I)**2 + DATA(I+1)**2
000274     600  CONTINUE
000276          IF (IOTERMS.EQ.0) GO TO 615
000277          WRITE(6,610) (DATA(I),I=1,IM2,2)
000311     610  FORMAT(//* SQUARE OF ABSOLUTE VALUE OF FAST FOURIER*
000311        1        * TRANSFORM OF SEQUENCE*/  (10 F7.1) )
000311     615  CONTINUE
              *
              *  MULTIPLY BY SIN(PI * J / P)**2  /  (PI * J / P)**2,
              *    WHICH IS THE DAMPING FACTOR
              *
000311          DO 620 I=3,IL2,2
000316          ADEN = PI * (I/2) / ILENGTH
000323          ANUM = SIN(ADEN)
000325          DATA(I) = DATA(I) * (ANUM/ADEN)**2
000330     620  CONTINUE
000336          IF (IOTERMS.EQ.0) GO TO 640
000337          WRITE(6,630) (DATA(I),I=1,IM2,2)
000351     630  FORMAT(//* ...AND MODIFIED BY DAMPING FACTOR*/
000351        1        (10F7.1) )
000351     640  CONTINUE
              *
              *  CALCULATE THE DB SPECTRAL INCREASE
              *
000351          DATAMAX = DATA(1)
000353          DO 650 I=3,IL2,2
000360          IF (DATAMAX .LT. DATA(I)) DATAMAX = DATA(I)
000364     650  CONTINUE
000367          SPECIN = 10.0 * ALOG10(DATAMAX / (ILENGTH+1) )
000401          IF (IOTERMS.EQ.0) GO TO 670
000402          WRITE(6,660) DATAMAX,SPECIN
000412     660  FORMAT(//* MAXIMUM DAMPED LINE =* F7.1//
000412        1        * SPECTRAL INCREASE =* F7.1 * DB*)
```

```
000412     670  CONTINUE
            *
            *
            *  OBTAIN ABSOLUTE VALUE OF REAL PART OF FAST FOURIER TRANSFORM
            *
            *       M2 = 2*IOTERMS
            *       DO 745 I=1,M2,2
            *       DATA(I) = ABS( DATA(I) )
            *745    CONTINUE
            *       WRITE(6,750) (DATA(I),I=1,M2,2)
            *750    FORMAT(//* ABSOLUTE VALUE OF REAL PART OF FAST FOURIER *
            *     1         *TRANSFORM OF SECUENCE*/ (10 F7.1) )
            *
            *
000412              CALL SECOND(T2)
000414              ELAPS = T2 - T1
000416              IF (IOTFRMS.EQ.0) GO TO 770
000423              WRITE(6,760) ELAPS
000430     760      FORMAT(//* ELAPSED CPU TIME TO OBTAIN *
000430      +       *FAST FOURIER TRANSFORM =* F10.3)
000430     770      CONTINUE
            *
000430              RETURN
000431              END
```

```
RUNX COMPILER (VER.2.3M)                    03/16/77. 20.24.42.

                     SUBROUTINE IFILIN(I,J,K,IERROR)
000007               DIMENSION A(7),B(7),FMT(7)
000007               DATA FMT/4H(I1),4H(I2),4H(I3),4H(I4),4H(I5),4H(I6),4H(I7)/
000007               DATA A/07777777777777777777,0777777777777777,077777777777777,
000007              1077777777777777,07777777777,077777777,0777777/
000007               DATA B/03300000000000000000,03333000000000000000,
000007              103333330000000000000,03333333300000000000,03333333333000000000,
000007              203333333333330000000,03333333333333000000/
000007               IF(I.GT.7.OR.I.LT.1.OR.J.GE.10**I)10,20
000025            10 IERROR = -1
000026               RETURN
000027            20 IERROR = 1
000030               ENCODE(I,FMT(I),N)J
000040               IF(I.EQ.1)105,25
000047            25 L = I-1
000051               DO 100 M=1,L
000051               IF(J.LT.10**M)200,100
000060           100 CONTINUE
000063           105 K = N
000064               RETURN
000065           200 NOS = I-M
000066               K = N.AND.A(NOS).OR.B(NOS)
000072               RETURN
000072               END
```

```
      SUBROUTINE FOURT (DATA,N,NDIM,ISIGN,IFORM)                        FFT   1
C     COOLEY-TUKEY FAST FOURIER TRANSFORM IN ANSI BASIC FORTRAN.        FFT   2
C     MULTI-DIMENSIONAL TRANSFORM, DIMENSIONS OF ARBITRARY SIZE,        FFT   3
C     COMPLEX OR REAL DATA.  N POINTS CAN BE TRANSFORMED IN TIME        FFT   4
C     PROPORTIONAL TO N*LOG(N), WHEREAS OTHER METHODS TAKE N**2 TIME.   FFT   5
C     ALSO, LESS ERROR IS BUILT UP.  WRITTEN BY NORMAN BRENNER, MIT,    FFT   6
C     FEBRUARY 1969.  SEE--IEEE AUDIO AND ELECTROACOUSTICS TRANSACTIONS FFT   7
C     OF JUNE 1967 AND JUNE 1969, SPECIAL FFT ISSUES.                   FFT   8
C                                                                       FFT   9
C     DIMENSION DATA(N(1),N(2),...),TRANSFORM(N(1),N(2),...),N(NDIM)    FFT  10
C     TRANSFORM(K1,K2,...) = SUM(DATA(J1,J2,...)*EXP(ISIGN*2*PI*SQRT(-1)FFT  11
C     *((J1-1)*(K1-1)/N(1)+(J2-1)*(K2-1)/N(2)+...))), SUMMED FOR ALL    FFT  12
C     J1 AND K1 FROM 1 TO N(1), J2 AND K2 FROM 1 TO N(2), ETC. FOR ALL  FFT  13
C     NDIM SUBSCRIPTS.  NDIM MUST BE POSITIVE AND EACH N(IDIM) MAY BE   FFT  14
C     ANY POSITIVE INTEGER, NONE OF WHOSE PRIME FACTORS IS GREATER THAN FFT  15
C     97.  DATA IS COMPLEX, REAL OR HALF-SIZE COMPLEX, AS IFORM = 1, 0  FFT  16
C     OR -1.  IN THE FIRST TWO CASES, DATA IS DIMENSIONED N(1) BY N(2)  FFT  17
C     BY ... BY N(NDIM) COMPLEX OR REAL, AS DESCRIBED IN THE N ARRAY.   FFT  18
C     IN THE LAST CASE, IT IS CONCEPTUALLY DIMENSIONED N(1) BY ... BY   FFT  19
C     N(NDIM) COMPLEX, BUT IN FACT IS ONLY N(1)/2+1 BY ... BY N(NDIM),  FFT  20
C     SINCE THE MISSING VALUES ARE KNOWN TO BE COMPLEX CONJUGATES OF    FFT  21
C     THOSE PRESENT.  SEE THE EXAMPLE.  ON RETURN, ARRAY DATA HAS BEEN  FFT  22
C     REFILLED WITH THE TRANSFORM VALUES, WHICH ARE COMPLEX, HALF-SIZE  FFT  23
C     COMPLEX OR REAL, RESPECTIVELY.  ISIGN IS +1 OR -1.  A +1 TRANS-   FFT  24
C     FORM FOLLOWED BY A -1 TRANSFORM (OR VICE VERSA) RETURNS NTOT      FFT  25
C     TIMES THE ORIGINAL VALUES TO DATA, WHERE NTOT = N(1)* ...         FFT  26
C     *N(NDIM).  IF IFORM = 0 OR -1, N(1) MUST BE EVEN AND ENOUGH       FFT  27
C     STORAGE RESERVED FOR THE HALF-SIZE COMPLEX ARRAY.                 FFT  28
C                                                                       FFT  29
C     EXAMPLE 1.  TRANSFORM OF A COMPLEX VECTOR 1960 LONG.              FFT  30
C     DIMENSION DATA(1960)                                              FFT  31
C     COMPLEX DATA                                                      FFT  32
C     CALL FOURT (DATA,1960,1,-1,+1)                                    FFT  33
C                                                                       FFT  34
C     EXAMPLE 2.  TRANSFORM OF A REAL VECTOR 1960 LONG.                 FFT  35
C     DIMENSION DATA(1960),TRAN(981)                                    FFT  36
C     COMPLEX TRAN                                                      FFT  37
```

```
            C      EQUIVALENCE (DATA,TRAN)
            C      CALL FOURT (DATA,1960,1,-1,0)
            C
            C      DIRECTLY IMPLEMENTING THE SUMMATION TAKES A TIME PROPORTIONAL TO
            C      NTOT**2, WHILE FAST FOURIER TRANSFORM TAKES TIME PROPORTIONAL
            C      TO NTOT*(SUM OF THE PRIME FACTORS OF NTOT).  THIS IS A TREMENDOUS
            C      SPEEDUP FOR LARGE NTOT.  FACTORS OF TWO AND THREE ARE ESPECIALLY
            C      FAST.  IF THE NUMBER OF POINTS IS PRIME, SIMPLY ADD ZERO DATA TO
            C      PAD OUT TO A HIGHLY COMPOSITE LENGTH.  FINALLY, REAL OR HALF-SIZE
            C      COMPLEX TRANSFORMS RUN TWICE AS FAST AS COMPLEX TRANSFORMS.
            C
            C      AN UPPER BOUND FOR THE RMS RELATIVE ERROR IS GIVEN BY GENTLEMAN
            C      AND SANDE -- 3 * 2**(-B) * SUM(F**1.5), WHERE 2**(-B) IS THE
            C      SMALLEST BIT IN THE FLOATING POINT FRACTION AND THE SUM IS OVER
            C      THE PRIME FACTORS OF NTOT.
            C
            C      IF THE INPUT DATA ARE A TIME SERIES, WITH INDEX J REPRESENTING
            C      A TIME (J-1)*DELTAT, THEN THE CORRESPONDING INDEX K IN THE
            C      TRANSFORM REPRESENTS THE FREQUENCY (K-1)*2*PI/(N*DELTAT), WHICH
            C      BY PERIODICITY, IS THE SAME AS FREQUENCY -(N-K+1)*2*PI/(N*DELTAT).
            C      THIS IS TRUE FOR N = EACH N(IDIM) INDEPENDENTLY.
            C
            C      MACHINE/COMPILER DEPENDENCIES--
            C      1.  REAL AND IMAGINARY PARTS MUST BE STORED ADJACENTLY (STANDARD).
            C      2.  ARRAYS ARE STORED LINEARLY WITH THE FIRST SUBSCRIPT INCREASING
            C      FASTEST (STANDARD).
            C      3.  FUNCTION SETLO, WHICH SETS THE LOW ORDER BIT OF THE DATA.
            C
000010             DIMENSION DATA(1),N(1),IFACT(32),IFSYM(32),IFCNT(10),WORK(2,97)
000010             NWORK=97
            C      MAXIMUM OF 32 PRIME FACTORS PER N(I), MAXIMUM OF 10 UNPAIRED
            C      FACTORS, MAXIMUM FACTOR OF 97
000011             IF (IFORM) 10,10,40
000012        10   IF (N(1)-2*(N(1)/2)) 20,40,20
000015        20   WRITE (6,30) IFORM,N(1)
000030        30   FORMAT (26H0ERROR IN FOURT.  IFORM = ,I2,41H (REAL OR HALF-SIZE CO
000030            $MPLEX), BUT N(1) = ,I5,37H IS NOT EVEN.  NO TRANSFORM WAS DONE.)
000030             RETURN
```

```
000031    40   NTOT=1                                                          FFT  76
000032         DO 50 IDIM=1,NDIM                                               FFT  77
000037    50   NTOT=NTOT*N(IDIM)                                               FFT  78
000044         NREM=NTOT                                                       FFT  79
000045         IF (IFORM) 60,70,70                                             FFT  80
000046    60   NREM=1                                                          FFT  81
000047         NTOT=(NTOT/N(1))*(N(1)/2+1)                                     FFT  82
          C    LOOP OVER ALL DIMENSIONS.                                       FFT  83
000056    70   DO 230 JDIM=1,NDIM                                              FFT  84
000060         IF (IFORM) 80,90,90                                             FFT  85
000061    80   IDIM=NDIM+1-JDIM                                                FFT  86
000064         GO TO 100                                                       FFT  87
000064    90   IDIM=JDIM                                                       FFT  88
000066         NREM=NREM/N(IDIM)                                               FFT  89
000072   100   NCURR=N(IDIM)                                                   FFT  90
000074         IF (IDIM-1) 110,110,140                                         FFT  91
000076   110   IF (IFORM) 120,130,140                                          FFT  92
          C    PRE-ADJUST A HALF-SIZE COMPLEX ARRAY                            FFT  93
000100   120   CALL FIXRL (DATA,N(1),NREM,ISIGN,IFORM)                         FFT  94
000104         NTOT=(NTOT/(N(1)/2+1))*N(1)                                     FFT  95
000115   130   NCURR=NCURR/2                                                   FFT  96
000117   140   IF (NCURR-1) 190,190,150                                        FFT  97
          C    FACTOR N(IDIM), AS 1960 = 2*2*2*3*7*7.                          FFT  98
000122   150   CALL FACTR (NCURR,IFACT,NFACT)                                  FFT  99
000125         IF (IFACT(NFACT)-NWORK) 180,180,160                             FFT 100
000133   160   WRITE (6,170) NWORK,IFACT(NFACT),IDIM,N(IDIM)                   FFT 101
000156   170   FORMAT (44HOERROR IN FOURT.  THE WORK ARRAY, OF LENGTH ,I3,     FFT 102
000156        $19H, IS TOO SMALL FOR ,I5,32H, THE LARGEST PRIME FACTOR OF N(,I1, FFT 103
000156        $4H) = ,I5,1H./23H NO TRANSFORM WAS DONE.)                       FFT 104
000156         RETURN                                                          FFT 105
          C    ARRANGE THE FACTORS SYMMETRICALLY FOR SIMPLER DIGIT REVERSAL.   FFT 106
000157   180   CALL SMFAC (IFACT,NFACT,ISYM,IFSYM,NFSYM,ICENT,IFCNT,NFCNT)     FFT 107
000167         NPREV=NTOT/(N(IDIM)*NREM)                                       FFT 108
          C    DIGIT REVERSE ON SYMMETRIC FACTORS, FOR EXAMPLE 2*7*6*7*2.      FFT 109
000200         CALL SYMRV (DATA,NPREV,NCURR,NREM,IFSYM,NFSYM)                  FFT 110
          C    DIGIT REVERSE THE ASYMMETRIC CENTER, FOR EXAMPLE, ON 6 = 2*3.   FFT 111
000203         CALL ASMRV (DATA,NPREV*ISYM,ICENT,ISYM*NREM,IFCNT,NFCNT)        FFT 112
          C    FOURIER TRANSFORM ON EACH FACTOR, FOR EXAMPLE, ON 2,7,2,3,7 AND 2.FFT 113
```

```
000217         CALL DFT (DATA,NPREV,NCURR,NREM,ISIGN,IFACT,WORK)           FFT 114
000231    190  IF (IFORM) 200,210,230                                      FFT 115
000236    200  NREM=NREM*N(IDIM)                                           FFT 116
000241         GO TO 230                                                   FFT 117
000242    210  IF (IDIM-1) 220,220,230                                     FFT 118
          C    POST-ADJUST THE TRANSFORM OF A REAL ARRAY, DISGUISED AS COMPLEX FFT 119
000245    220  CALL FIXRL (DATA,N(1),NREM,ISIGN,IFORM)                     FFT 120
000251         NTOT=NTOT/N(1)*(N(1)/2+1)                                   FFT 121
000262    230  CONTINUE                                                    FFT 122
000265         RETURN                                                      FFT 123
000265         END                                                         FFT 124
```

```
RUNX COMPILER (VER.2.3M)                03/16/77. 20.27.08.

                 SUBROUTINE FACTR (N,IFACT,NFACT)                              FAC   1
           C     FACTOR N INTO ITS PRIME FACTORS, NFACT IN NUMBER.  FOR EXAMPLE,  FAC   2
           C     FOR N = 1960, NFACT = 6 AND IFACT(IF) = 2, 2, 2, 5, 7 AND 7.     FAC   3
000006           DIMENSION IFACT(1)                                             FAC   4
000006           IF=0                                                           FAC   5
000006           NPART=N                                                        FAC   6
000010           DO 50 ID=1,N,2                                                 FAC   7
000011           IDIV=ID                                                        FAC   8
000012           IF (ID-1) 10,10,20                                             FAC   9
000014     10    IDIV=2                                                         FAC  10
000015     20    IQUOT=NPART/IDIV                                               FAC  11
000020           IF (NPART-IDIV*IQUOT) 40,30,40                                 FAC  12
000023     30    IF=IF+1                                                        FAC  13
000025           IFACT(IF)=IDIV                                                 FAC  14
000027           NPART=IQUOT                                                    FAC  15
000031           GO TO 20                                                       FAC  16
000031     40    IF (IQUOT-IDIV) 60,60,50                                       FAC  17
000034     50    CONTINUE                                                       FAC  18
000037     60    IF (NPART-1) 80,80,70                                          FAC  19
000042     70    IF=IF+1                                                        FAC  20
000044           IFACT(IF)=NPART                                                FAC  21
000045     80    NFACT=IF                                                       FAC  22
000047           RETURN                                                         FAC  23
000050           END                                                            FAC  24
```

```
RUNX COMPILER (VER.2.3M)                    03/16/77. 20.27.08.

                 SUBROUTINE SMFAC (IFACT,NFACT,ISYM,IFSYM,NFSYM,ICENT,IFCNT,NFCNT) SMF  1
          C      REARRANGE THE PRIME FACTORS OF N INTO A SQUARE AND A NON-         SMF  2
          C      SQUARE.  N = ISYM*ICENT*ISYM, WHERE ICENT IS SQUARE-FREE.         SMF  3
          C      ISYM = IFSYM(1)*...*IFSYM(NFSYM), EACH A PRIME FACTOR.            SMF  4
          C      ICENT = IFCNT(1)*...*IFCNT(NFCNT), EACH A PRIME FACTOR.           SMF  5
          C      FOR EXAMPLE, N = 1960 = 14*10*14.  THEN ISYM = 14, ICENT = 10,    SMF  6
          C      NFSYM = 2, NFCNT = 2, NFACT = 6, IFSYM(IFS) = 2, 7, IFCNT(IFC) =  SMF  7
          C      2, 5 AND IFACT(IF) = 2, 7, 2, 5, 7, 2.                            SMF  8
000013           DIMENSION IFSYM(1), IFCNT(1), IFACT(1)                            SMF  9
000013           ISYM=1                                                            SMF 10
000013           ICENT=1                                                           SMF 11
000014           IFS=0                                                             SMF 12
000015           IFC=0                                                             SMF 13
000015           IF=1                                                              SMF 14
000017     10    IF (IF-NFACT) 20,40,50                                            SMF 15
000021     20    IF (IFACT(IF)-IFACT(IF+1)) 40,30,40                               SMF 16
000024     30    IFS=IFS+1                                                         SMF 17
000026           IFSYM(IFS)=IFACT(IF)                                              SMF 18
000031           ISYM=IFACT(IF)*ISYM                                               SMF 19
000034           IF=IF+2                                                           SMF 20
000035           GO TO 10                                                          SMF 21
000036     40    IFC=IFC+1                                                         SMF 22
000040           IFCNT(IFC)=IFACT(IF)                                              SMF 23
000043           ICENT=IFACT(IF)*ICENT                                             SMF 24
000046           IF=IF+1                                                           SMF 25
000047           GO TO 10                                                          SMF 26
000050     50    NFSYM=IFS                                                         SMF 27
000051           NFCNT=IFC                                                         SMF 28
000052           NFSM2=2*NFSYM                                                     SMF 29
000053           NFACT=2*NFSYM+NFCNT                                               SMF 30
000055           IF (NFCNT) 80,80,60                                               SMF 31
000056     60    NFSM2=NFSM2+1                                                     SMF 32
000060           IFSYM(NFSYM+1)=ICENT                                              SMF 33
000061           DO 70 IFC=1,NFCNT                                                 SMF 34
000063           IF=NFSYM+IFC                                                      SMF 35
000064     70    IFACT(IF)=IFCNT(IFC)                                              SMF 36
000072     80    IF (NFSYM) 110,110,90                                             SMF 37
```

```
RUNX COMPILER (VER.2.3M)              03/16/77. 20.27.08.         SMFAC

000074     90    DO 100 IFS=1,NFSYM                                 SMF   38
000076           IFSCJ=NFSM2+1-IFS                                  SMF   39
000100           IFSYM(IFSCJ)=IFSYM(IFS)                            SMF   40
000103           IFACT(IFS)=IFSYM(IFS)                              SMF   41
000106           IFCNJ=NFACT+1-IFS                                  SMF   42
000110    100    IFACT(IFCNJ)=IFSYM(IFS)                            SMF   43
000114    110    NFSYM=NFSM2                                        SMF   44
000115           RETURN                                             SMF   45
000116           END                                                SMF   46
```

```
                SUBROUTINE SYMRV (DATA,NPREV,N,NREM,IFACT,NFACT)
          C     PERMUTE THE DATA ARRAY BY REVERSING THE DIGITS OF ONE INDEX.
          C     DIMENSION DATA(NPREV,N,NREM)
          C     REPLACE DATA(I1,I2,I3) BY DATA(I1,I2REV,I3) FOR ALL I1 FROM 1 TO
          C     NPREV, I2 FROM 1 TO N AND I3 FROM 1 TO NREM.  I2REV-1 IS THE
          C     INTEGER WHOSE DIGIT REPRESENTATION IN THE MULTI-RADIX NOTATION
          C     OF FACTORS IFACT(IF) IS THE REVERSE OF THE REPRESENTATION OF I2-1.
          C     FOR EXAMPLE, IF ALL IFACT(IF) = 2, I2-1 = 11001, I2REV-1 = 10011.
          C     THE FACTORS MUST BE SYMMETRICALLY ARRANGED, I.E., IFACT(IF) =
          C     IFACT(NFACT+1-IF).  THEN, PAIRWISE EXCHANGE CAN BE DONE.
)00011          DIMENSION DATA(1), IFACT(1)
)00011          IF (NFACT-1) 80,80,10
)00013     10   IP0=2
)00014          IP1=IP0*NPREV
)00016          IP4=IP1*N
)00020          IP5=IP4*NPEM
)00022          I4REV=1
)00023          DO 70 I4=1,IP4,IP1
)00025          IF (I4-I4REV) 20,40,40
)00027     20   I1MAX=I4+IP1-IP0
)00032          DO 30 I1=I4,I1MAX,IP0
)00033          DO 30 I5=I1,IP5,IP4
)00034          I5REV=I4REV+I5-I4
)00036          TEMPR=DATA(I5)
)00040          TEMPI=DATA(I5+1)
)00042          DATA(I5)=DATA(I5REV)
)00045          DATA(I5+1)=DATA(I5REV+1)
)00046          DATA(I5REV)=TEMPR
)00050     30   DATA(I5REV+1)=TEMPI
)00057     40   IP3=IP4
)00061          DO 60 IF=1,NFACT
)00062          IP2=IP3/IFACT(IF)
)00066          I4REV=I4REV+IP2
)00067          IF (I4REV-IP3) 70,70,50
)00071     50   I4REV=I4REV-IP3
)00073     60   IP3=IP2
)00076     70   CONTINUE
```

```
RUNX COMPILER (VER.2.3M)                03/16/77. 20.27.08.        SYMRV

000101     80    RETURN                                                 SYM  38
000102           END                                                    SYM  39-
```

```
JNX COMPILER (VER.2.3M)                    03/16/77. 20.27.08.

                SUBROUTINE ASMRV (DATA,NPREV,N,NREM,IFACT,NFACT)                 ASM   1
         C      PERMUTE THE DATA ARRAY BY REVERSING THE DIGITS OF ONE INDEX.     ASM   2
         C      THE OPERATION IS THE SAME AS IN SYMRV, EXCEPT THAT THE FACTORS   ASM   3
         C      NEED NOT BE SYMMETRICALLY ARRANGED, I.E., GENERALLY IFACT (IF)   ASM   4
         C      NOT = IFACT(NFACT+1-IF).  NO WORK ARRAY IS NEEDED BY USING SANDESSASM   5
         C      METHOD OF CYCLE TRACING, WHICH TRANSFORMS THE PERMUTATION INTO A ASM   6
         C      SET OF PAIR INTERCHANGES.  THE DATA ARE MARKED IN THE LOWEST BIT ASM   7
         C      TO KEEP TRACK OF THE INTERCHANGES.                               ASM   8
00011           DIMENSION DATA(1), IFACT(1), MOD(10)                             ASM   9
00011           IF (NFACT-1) 140,140,10                                          ASM  10
00013     10    IP0=2                                                            ASM  11
00014           IP1=IP0*NPREV                                                    ASM  12
00016           IP2=IP1*N                                                        ASM  13
00020           IP3=IP2*NREM                                                     ASM  14
00022           MULT=N/IFACT(1)                                                  ASM  15
         C      COMPUTE THE MODULOES.                                            ASM  16
00025           IPROD=MULT*IP1                                                   ASM  17
00027           INVPR=IPROD                                                      ASM  18
00031           DO 20 IF=2,NFACT                                                 ASM  19
00032           IPROD=IPROD*IFACT(IF-1)                                          ASM  20
00035           INVPR=INVPR/IFACT(IF)                                            ASM  21
00041     20    MOD(IF)=IPROD-INVPR                                              ASM  22
         C      MARK THE DATA INITIALLY                                          ASM  23
00044           DO 30 I2MIN=1,IP2,IP1                                            ASM  24
00046     30    DATA(I2MIN)=SETLO(DATA(I2MIN),1)                                 ASM  25
00061           DO 130 I2MIN=1,IP2,IP1                                           ASM  26
         C      IF THE DATUM IS UNMARKED, IT HAS BEEN PERMUTED TO ITS NEW PLACE-- ASM  27
         C      EXIT.                                                            ASM  28
00063           IF (DATA(I2MIN)-SETLO(DATA(I2MIN),0)) 40,130,40                  ASM  29
00075     40    DATA(I2MIN)=SETLO(DATA(I2MIN),0)                                 ASM  30
00106           I2=I2MIN                                                         ASM  31
         C      COMPUTE THE ELEMENTS OF A SUBCYCLE OF THE PERMUTATION AND        ASM  32
         C      INTERCHANGE THE DATA PAIRWISE TO ACCOMPLISH IT.                  ASM  33
00110     50    I2REV=(I2-1)*MULT+1                                              ASM  34
00113           IF=NFACT                                                         ASM  35
00115           GO TO 70                                                         ASM  36
00115     60    I2REV=I2REV-MOD(IF)*((I2REV-1)/MOD(IF))                          ASM  37
```

```
UNX COMPILER (VER.2.3M)                03/16/77. 20.27.08.          ASMRV

00124          IF=IF-1                                                    ASM  38
00126     70   IF (IF-2) 80,80,60                                         ASM  39
00131     80   IOUOT=(I2REV-1)/MOD(2)                                     ASM  40
00135          IF (IOUOT-IFACT(2)) 100,90,90                              ASM  41
00140     90   IOUOT=IOUOT-1                                              ASM  42
00142    100   I2REV=I2REV-MOD(2)*IOUOT                                   ASM  43
         C     EXIT IF THE SUBCYCLE HAS CLOSED ON ITSELF                  ASM  44
00145          IF (I2REV-I2MIN) 110,130,110                               ASM  45
00147    110   DATA(I2REV)=SETLO(DATA(I2REV),0)                           ASM  46
00160          I1MAX=I2+IP1-IP0                                           ASM  47
00163          DO 120 I1=I2,I1MAX,IP0                                     ASM  48
00164          DO 120 I3=I1,IP3,IP2                                       ASM  49
00165          I3REV=I3+I2REV-I2                                          ASM  50
00167          TEMPR=DATA(I3)                                             ASM  51
00171          TEMPI=DATA(I3+1)                                           ASM  52
00173          DATA(I3)=DATA(I3REV)                                       ASM  53
00176          DATA(I3+1)=DATA(I3REV+1)                                   ASM  54
00177          DATA(I3REV)=TEMPR                                          ASM  55
00201    120   DATA(I3REV+1)=TEMPI                                        ASM  56
00210          I2=I2REV                                                   ASM  57
         C     GO BACK TO MOVE THE NEXT ELEMENT IN THE SUBCYCLE           ASM  58
00211          GO TO 50                                                   ASM  59
00212    130   CONTINUE                                                   ASM  60
00215    140   RETURN                                                     ASM  61
00216          END                                                        ASM  62-
```

UNX COMPILER (VER.2.3M) 03/16/77. 20.27.08.

```
                FUNCTION SETLO(IDATA,IBIT)
          C     IDATA IS A REAL OR EXTENDED FLOATING-POINT NUMBER DISGUISED AS
          C     AN INTEGER ARRAY.  IBIT IS 0 OR 1.  BY ANY MEANS AVAILABLE
          C     SET THE LOW ORDER BIT OF THE FRACTION OF IDATA TO IBIT.  THE
          C     METHOD MUST BE TAILORED TO EACH MACHINE AND DATA LENGTH, BUT IT
          C     NEED NOT BE EFFICIENT, AS THIS ROUTINE IS CALLED INFREQUENTLY.
          C     THE RESULT IS RETURNED UNDER THE FLOATING POINT NAME SETLO.
          C     THIS ROUTINE IS NOT CALLED BY FOURT IF THE LENGTH OF ARRAY DATA
          C     IS A POWER OF TWO, OR A PERFECT SQUARE, OR A SQUARE TIMES A
          C     PRIME FACTOR.
          C     FOR THE TRW CDC COMPUTER REAL*4--
00005           DIMENSION IDATA(1),ISETL(1)
00005           EQUIVALENCE (ISETL(1),SETL)
00005           ISETL(1)=2*(IDATA(1)/2)+IBIT
00007           SETLO=SETL
          C     FOR THE TRW CDC COMPUTER REAL*8--
          C     DIMENSION IDATA(2),ISETL(2)
          C     DOUBLE PRECISION SETLO,SETL
          C     EQUIVALENCE (ISETL(1),SETL)
          C     ISETL(1)=IDATA(1)
          C     ISETL(2)=2*(IDATA(2)/2)+IBIT
          C     SETLO=SETL
          C     THE TECHNIQUE DEMONSTRATED WILL NOT WORK IF THE EXPONENT OCCUPIES
          C     THE RIGHTMOST BITS OF THE DATUM, AS FOR IBM 1130 REGULAR FLOATING
          C     POINT.
00011           RETURN
00011           END
```

```
UNX COMPILER (VER.2.3M)                   03/16/77. 20.27.08.

                SUBROUTINE DFT (DATA,NPREV,N,NREM,ISIGN,IFACT,WORK)               DFT   1
          C     DISCRETE FOURIER TRANSFORM OF LENGTH N.  IN PLACE COOLEY-TUKEY     DFT   2
          C     METHOD, DIGIT-REVERSED TO NORMAL ORDER, SANDE-TUKEY PHASE SHIFTS.  DFT   3
          C     DIMENSION DATA(NPREV,N,NREM)                                       COO   4
          C     COMPLEX DATA                                                       COO   5
          C     DATA(I1,J2,I3) = SUM(DATA(I1,I2,I3)*EXP(ISIGN*2*PI*I*((I2-1)*      COO   6
          C     (J2-1)/N))), SUMMED OVER I2 = 1 TO N FOR ALL I1 FROM 1 TO NPREV,   COO   7
          C     J2 FROM 1 TO N AND I3 FROM 1 TO NREM.  THE FACTORS OF N ARE GIVEN  COO   8
          C     IN ANY ORDER IN ARRAY IFACT.  FACTORS OF TWO ARE DONE IN PAIRS     COO   9
          C     AS MUCH AS POSSIBLE (FOURIER TRANSFORM OF LENGTH FOUR), FACTORS OFCOO  10
          C     THREE ARE DONE SEPARATELY, AND ALL FACTORS FIVE OR HIGHER          COO  11
          C     ARE DONE BY GOERTZELSS ALGORITHM.                                  COO  12
00012           DIMENSION DATA(1), WORK(1), IFACT(1)                               COO  13
          C     REMOVE THE NEXT THREE CARDS IF NO DOUBLE PRECISION FEATURE EXISTS.COO  14
          C     THEY SERVE TO INCREASE ACCURACY OF THE SINE AND COSINE VALUES.
00012           DOUBLE PRECISION TWOPI,THETA,SIN,SINTH,ROOTR,ROOTI,WSTPR,WSTPI,WR,COO  15
00012          $WI,TEMP                                                            COO  16
00012           SIN(THETA)=DSIN(THETA)                                             COO  17
00027           TWOPI=6.2831853071795865*FLOAT(ISIGN)                              COO  18
00032           IP0=2                                                              COO  19
00033           IP1=IP0*NPREV                                                      COO  20
00035           IP4=IP1*N                                                          COO  21
00037           IP5=IP4*NREM                                                       COO  22
00041           IF=0                                                               COO  23
00042           IP2=IP1                                                            COO  24
00044      10   IF (IP2-IP4) 20,240,240                                            COO  25
00047      20   IF=IF+1                                                            COO  26
00051           IFCUR=IFACT(IF)                                                    COO  27
00053           IF (IFCUR-2) 60,30,60                                              COO  28
00055      30   IF (4*IP2-IP4) 40,40,60                                            COO  29
00060      40   IF (IFACT(IF+1)-2) 60,50,60                                        COO  30
00063      50   IF=IF+1                                                            COO  31
00065           IFCUR=4                                                            COO  32
00066      60   IP3=IP2*IFCUR                                                      COO  33
00070           THETA=TWOPI/FLOAT(IFCUR)                                           COO  34
00104           SINTH=SIN(THETA/2.)                                                COO  35
00121           ROOTR=-2.*SINTH*SINTH                                              COO  36
```

```
00131          ROOTI=SIN(THETA)                                    C00  37
00135          THETA=TWOPI/FLOAT(IP3/IP1)                          C00  38
00153          SINTH=SIN(THETA/2.)                                 C00  39
00170          WSTPR=-2.*SINTH*SINTH                               C00  40
00200          WSTPI=SIN(THETA)                                    C00  41
00204          WR=1.                                               C00  42
00206          WI=0.                                               C00  43
00210          DO 230 I2=1,IP2,IP1                                 C00  44
00211          IF (IFCUR-4) 70,70,210                              C00  45
00213     70   IF ((I2-1)*(IFCUR-2)) 240,90,80                     C00  46
00220     80   W2R=WR*WR-WI*WI                                     C00  47
00233          W2I=2.*WR*WI                                        C00  48
00243          W3R=W2R*WR-W2I*WI                                   C00  49
00257          W3I=W2R*WI+W2I*WR                                   C00  50
00273     90   I1MAX=I2+IP1-IP0                                    C00  51
00276          DO 200 I1=I2,I1MAX,IP0                              C00  52
00277          DO 200 I5=I1,IP5,IP3                                C00  53
00300          J0=I5                                               C00  54
00301          J1=J0+IP2                                           C00  55
00302          J2=J1+IP2                                           C00  56
00303          J3=J2+IP2                                           C00  57
00304          IF (I2-1) 140,140,100                               C00  58
00307    100   IF (IFCUR-3) 130,120,110                            C00  59
        C      APPLY THE PHASE SHIFT FACTORS                       C00  60
00312    110   TEMPR=DATA(J3)                                      C00  61
00315          DATA(J3)=W3R*TEMPR-W3I*DATA(J3+1)                   C00  62
00321          DATA(J3+1)=W3R*DATA(J3+1)+W3I*TEMPR                 C00  63
00324          TEMPR=DATA(J2)                                      C00  64
00326          DATA(J2)=WR*TEMPR-WI*DATA(J2+1)                     C00  65
00343          DATA(J2+1)=WR*DATA(J2+1)+WI*TEMPR                   C00  66
00357          TEMPR=DATA(J1)                                      C00  67
00361          DATA(J1)=W2R*TEMPR-W2I*DATA(J1+1)                   C00  68
00365          DATA(J1+1)=W2R*DATA(J1+1)+W2I*TEMPR                 C00  69
00370          GO TO 140                                           C00  70
00370    120   TEMPR=DATA(J2)                                      C00  71
00373          DATA(J2)=W2R*TEMPR-W2I*DATA(J2+1)                   C00  72
00377          DATA(J2+1)=W2R*DATA(J2+1)+W2I*TEMPR                 C00  73
00402    130   TEMPR=DATA(J1)                                      C00  74
```

```
000405          DATA(J1)=WR*TEMPR-WI*DATA(J1+1)
000422          DATA(J1+1)=WR*DATA(J1+1)+WI*TEMPR
000437    140   IF (IFCUR-3) 150,160,170
         C      DO A FOURIER TRANSFORM OF LENGTH TWO
000442    150   TEMPR=DATA(J1)
000444          TEMPI=DATA(J1+1)
000446          DATA(J1)=DATA(J0)-TEMPR
000452          DATA(J1+1)=DATA(J0+1)-TEMPI
000455          DATA(J0)=DATA(J0)+TEMPR
000457          DATA(J0+1)=DATA(J0+1)+TEMPI
000461          GO TO 200
         C      DO A FOURIER TRANSFORM OF LENGTH THREE
000461    160   SUMR=DATA(J1)+DATA(J2)
000465          SUMI=DATA(J1+1)+DATA(J2+1)
000470          TEMPR=DATA(J0)-.5*SUMR
000474          TEMPI=DATA(J0+1)-.5*SUMI
000477          DATA(J0)=DATA(J0)+SUMR
000502          DATA(J0+1)=DATA(J0+1)+SUMI
000503          DIFR=ROOTI*(DATA(J2+1)-DATA(J1+1))
000514          DIFI=ROOTI*(DATA(J1)-DATA(J2))
000525          DATA(J1)=TEMPR+DIFR
000531          DATA(J1+1)=TEMPI+DIFI
000533          DATA(J2)=TEMPR-DIFR
000537          DATA(J2+1)=TEMPI-DIFI
000540          GO TO 200
         C      DO A FOURIER TRANSFORM OF LENGTH FOUR (FROM BIT REVERSED ORDER)
000541    170   T0R=DATA(J0)+DATA(J1)
000545          T0I=DATA(J0+1)+DATA(J1+1)
000550          T1R=DATA(J0)-DATA(J1)
000553          T1I=DATA(J0+1)-DATA(J1+1)
000556          T2R=DATA(J2)+DATA(J3)
000562          T2I=DATA(J2+1)+DATA(J3+1)
000565          T3R=DATA(J2)-DATA(J3)
000570          T3I=DATA(J2+1)-DATA(J3+1)
000573          DATA(J0)=T0R+T2R
000577          DATA(J0+1)=T0I+T2I
000601          DATA(J2)=T0R-T2R
000605          DATA(J2+1)=T0I-T2I
```

```
000606           IF (ISIGN) 180,180,190                                         COO 113
000607     180   T3R=-T3R                                                       COO 114
000610           T3I=-T3I                                                       COO 115
000612     190   DATA(J1)=T1R-T3I                                               COO 116
000616           DATA(J1+1)=T1I+T3R                                             COO 117
000620           DATA(J3)=T1R+T3I                                               COO 118
000624           DATA(J3+1)=T1I-T3R                                             COO 119
000625     200   CONTINUE                                                       COO 120
000632           GO TO 220                                                      COO 121
         C       DO A FOURIER TRANSFORM OF LENGTH FIVE OR MORE                  COO 122
000633     210   CALL GOERT (DATA(I2),NPREV,IP2/IP1,IFCUR,IF5/IP3,WORK,WR,WI,ROOTR,COO 123
000633          $ROOTI)                                                         COO 124
000655     220   TEMP=WR                                                        COO 125
000660           WR=WR*WSTPR-WI*WSTPI+WR                                        COO 126
000701     230   WI=WI*WSTPR+TEMP*WSTPI+WI                                      COO 127
000730           IP2=IP3                                                        COO 128
000731           GO TO 10                                                       COO 129
000732     240   RETURN                                                         COO 130
000732           END                                                            COO 131
```

```
RUNX COMPILER (VER.2.3M)                03/16/77. 20.27.08.

              SUBROUTINE GOERT(DATA,NPREV,IPROD,IFACT,IREM,WORK,WMINR,WMINI,    GOE   1
             $ ROOTR,ROOTI)                                                     GOE   2
          C   PHASE-SHIFTED FOURIER TRANSFORM OF LENGTH IFACT BY THE GOERTZEL   GOE   3
          C   ALGORITHM.  IFACT MUST BE ODD AND AT LEAST FIVE.  FURTHER SPEED   GOE   4
          C   IS GAINED BY COMPUTING TWO TRANSFORM VALUES AT THE SAME TIME.     GOE   5
          C   DIMENSION DATA(NPREV,IPROD,IFACT,IREM)                            GOE   6
          C   DATA(I1,1,J3,I5) = SUM(DATA(I1,1,I3,I5) * W**(I3-1)), SUMMED      GOE   7
          C   OVER I3 = 1 TO IFACT FOR ALL I1 FROM 1 TO NPREV, J3 FROM 1 TO     GOE   8
          C   IFACT AND I5 FROM 1 TO IREM.                                      GOE   9
          C   W = WMIN * EXP(ISIGN*2*PI*I*(J3-1)/IFACT).                        GOE  10
          C   REMOVE THE NEXT CARD IF NO DOUBLE PRECISION FEATURE EXISTS.       GOE  11
          C   IT SERVES TO INCREASE ACCURACY OF THE SINE AND COSINE VALUES.     GOE  111
000015        DOUBLE PRECISION WMINR,WMINI,ROOTR,ROOTI,TEMP                     GOE  12
000015        DIMENSION DATA(1), WORK(1)                                        GOE  13
000015        IP0=2                                                             GOE  14
000015        IP1=IP0*NPREV                                                     GOE  15
000017        IP2=IP1*IPROD                                                     GOE  16
000021        IP3=IP2*IFACT                                                     GOE  17
000023        IP5=IP3*IREM                                                      GOE  18
000025        IF (WMINI) 10,40,10                                               GOE  19
          C   APPLY THE PHASE SHIFT FACTORS                                     GOE  20
000030    10  WR=WMINR                                                          GOE  21
000033        WI=WMINI                                                          GOE  22
000035        I3MIN=1+IP2                                                       GOE  23
000037        DO 30 I3=I3MIN,IP3,IP2                                            GOE  24
000040        I1MAX=I3+IP1-IP0                                                  GOE  25
000043        DO 20 I1=I3,I1MAX,IP0                                             GOE  26
000044        DO 20 I5=I1,IP5,IP3                                               GOE  27
000045        TEMPR=DATA(I5)                                                    GOE  28
000047        DATA(I5)=WR*TEMPR-WI*DATA(I5+1)                                   GOE  29
000054    20  DATA(I5+1)=WR*DATA(I5+1)+WI*TEMPR                                 GOE  30
000063        TEMPR=WR                                                          GOE  31
000064        WR=WMINR*TEMPR-WMINI*WI                                           GOE  32
000101    30  WI=WMINR*WI+WMINI*TEMPR                                           GOE  33
000121    40  DO 90 I1=1,IP1,IP0                                                GOE  34
000123        DO 90 I5=I1,IP5,IP3                                               GOE  35
          C   STRAIGHT SUMMATION FOR THE FIRST TERM                             GOE  36
```

```
RUNX COMPILER (VER.2.3M)                03/16/77. 20.27.08.        GOERT

000124           SUMR=0.                                                              GOE  37
000125           SUMI=0.                                                              GOE  38
000125           I3MAX=I5+IP3-IP2                                                     GOE  39
000130           DO 50 I3=I5,I3MAX,IP2                                                GOE  40
000132           SUMR=SUMR+DATA(I3)                                                   GOE  41
000135      50   SUMI=SUMI+DATA(I3+1)                                                 GOE  42
000141           WORK(1)=SUMR                                                         GOE  43
000142           WORK(2)=SUMI                                                         GOE  44
000144           WR=ROOTR+1.                                                          GOE  45
000153           WI=ROOTI                                                             GOE  46
000155           IWMIN=1+IPO                                                          GOE  47
000157           IWMAX=IPO*((IFACT+1)/2)-1                                            GOE  48
000163           DO 80 IWORK=IWMIN,IWMAX,IPO                                          GOE  49
000165           TWOWR=WR+WR                                                          GOE  50
000166           I3=I3MAX                                                             GOE  51
000170           OLDSR=0.                                                             GOE  52
000170           OLDSI=0.                                                             GOE  53
000171           SUMR=DATA(I3)                                                        GOE  54
000174           SUMI=DATA(I3+1)                                                      GOE  55
000175           I3=I3-IP2                                                            GOE  56
000177      60   TEMPR=SUMR                                                           GOE  57
000200           TEMPI=SUMI                                                           GOE  58
000202           SUMR=TWOWR*SUMR-OLDSR+DATA(I3)                                       GOE  59
000206           SUMI=TWOWR*SUMI-OLDSI+DATA(I3+1)                                     GOE  60
000212           OLDSR=TEMPR                                                          GOE  61
000213           OLDSI=TEMPI                                                          GOE  62
000215           I3=I3-IP2                                                            GOE  63
000217           IF (I3-I5) 70,70,60                                                  GOE  64
           C     IN A FOURIER TRANSFORM THE W CORRESPONDING TO THE POINT AT K         GOE  65
           C     IS THE CONJUGATE OF THAT AT IFACT-K (THAT IS, EXP(TWOPI*I*           GOE  66
           C     K/IFACT) = CONJ(EXP(TWOPI*I*(IFACT-K)/IFACT))).  SINCE THE           GOE  67
           C     MAIN LOOP OF GOERTZELS ALGORITHM IS INDIFFERENT TO THE IMAGINARY     GOE  68
           C     PART OF W, IT NEED BE SUPPLIED ONLY AT THE END.                      GOE  69
000221      70   TEMPR=-WI*SUMI                                                       GOE  70
000223           TEMPI=WI*SUMR                                                        GOE  71
000225           SUMR=WR*SUMR-OLDSR+DATA(I3)                                          GOE  72
000231           SUMI=WR*SUMI-OLDSI+DATA(I3+1)                                        GOE  73
000235           WORK(IWORK)=SUMR+TEMPR                                               GOE  74
```

```
RUNX COMPILER (VER.2.3M)                03/16/77. 20.27.08.          GOERT

000240        WORK(IWORK+1)=SUMI+TEMPI
000242        IWCNJ=IPO*(IFACT+1)-IWORK                              GOE  75
000246        WORK(IWCNJ)=SUMR-TEMPR                                 GOE  76
000252        WORK(IWCNJ+1)=SUMI-TEMPI                               GOE  77
       C      SINGLETONS RECURSION, FOR ACCURACY AND SPEED.          GOE  78
000254        TEMP=WR                                                GOE  79
000256        WR=WR*ROOTR-WI*ROOTI+WR                                GOE  80
000276     80 WI=WI*ROOTR+TEMP*ROOTI+WI                              GOE  81
000321        IWORK=1                                                GOE  82
000322        DO 90 I3=I5,I3MAX,IP2                                  GOE  83
000324        DATA(I3)=WORK(IWORK)                                   GOE  84
000327        DATA(I3+1)=WORK(IWORK+1)                               GOE  85
000331     90 IWORK=IWORK+IPO                                        GOE  86
000342        RETURN                                                 GOE  87
000342        END                                                    GOE  88
                                                                     GOE  89-
```

```
                 SUBROUTINE FIXRL (DATA,N,NREM,ISIGN,IFORM)                      FIX   1
           C     FOR IFORM = 0, CONVERT THE TRANSFORM OF A DOUBLED-UP REAL ARRAY, FIX   2
           C     CONSIDERED COMPLEX, INTO ITS TRUE TRANSFORM.  SUPPLY ONLY THE   FIX   3
           C     FIRST HALF OF THE COMPLEX TRANSFORM, AS THE SECOND HALF HAS     FIX   4
           C     CONJUGATE SYMMETRY.  FOR IFORM = -1, CONVERT THE FIRST HALF     FIX   5
           C     OF THE TRUE TRANSFORM INTO THE TRANSFORM OF A DOUBLED-UP REAL   FIX   6
           C     ARRAY.  N MUST BE EVEN.                                         FIX   7
           C     USING COMPLEX NOTATION AND SUBSCRIPTS STARTING AT ZERO, THE     FIX   8
           C     TRANSFORMATION IS--                                             FIX   9
           C     DIMENSION DATA(N/2,NREM)                                        FIX  10
           C     ZSTP = EXP(ISIGN*2*PI*I/N)                                      FIX  11
           C     DO 10 I2=0,NREM-1                                               FIX  12
           C     DATA(0,I2) = CONJ(DATA(0,I2))*(1+I)                             FIX  13
           C     DO 10 I1=1,N/4                                                  FIX  14
           C     Z = (1+(2*IFORM+1)*I*ZSTP**I1)/2                                FIX  15
           C     I1CNJ = N/2-I1                                                  FIX  16
           C     DIF = DATA(I1,I2)-CONJ(DATA(I1CNJ,I2))                          FIX  17
           C     TEMP = Z*DIF                                                    FIX  18
           C     DATA(I1,I2) = (DATA(I1,I2)-TEMP)*(1-IFORM)                      FIX  19
           C 10  DATA(I1CNJ,I2) = (DATA(I1CNJ,I2)+CONJ(TEMP))*(1-IFORM)          FIX  20
           C     IF I1=I1CNJ, THE CALCULATION FOR THAT VALUE COLLAPSES INTO      FIX  21
           C     A SIMPLE CONJUGATION OF DATA(I1,I2).                            FIX  22
           C     REMOVE THE NEXT TWO CARDS IF NO DOUBLE PRECISION FEATURE EXISTS. FIX  23
           C     THEY SERVE TO INCREASE ACCURACY OF THE SINE AND COSINE VALUES.  FIX  231
000010           DOUBLE PRECISION TWOPI,THETA,SIN,SINTH,ZSTPR,ZSTPI,ZR,ZI,TEMP   FIX  24
000010           SIN(THETA)=DSIN(THETA)                                          FIX  25
000024           DIMENSION DATA(1)                                               FIX  26
000024           TWOPI=6.2831853071795865*FLOAT(ISIGN)                           FIX  27
000027           IP0=2                                                           FIX  28
000030           IP1=IP0*(N/2)                                                   FIX  29
000033           IP2=IP1*NREM                                                    FIX  30
000035           IF (IFORM) 10,70,70                                             FIX  31
           C     PACK THE REAL INPUT VALUES (TWO PER COLUMN)                     FIX  32
000037      10   J1=IP1+1                                                        FIX  33
000041           DATA(2)=DATA(J1)                                                FIX  34
000043           IF (NREM-1) 70,70,20                                            FIX  35
000045      20   J1=J1+IP0                                                       FIX  36
```

```
000047          I2MIN=IP1+1                                         FIX  37
000051          DO 60 I2=I2MIN,IP2,IP1                              FIX  38
000052          DATA(I2)=DATA(J1)                                   FIX  39
000055          J1=J1+IP0                                           FIX  40
000056          IF (N-2) 50,50,30                                   FIX  41
000061    30    I1MIN=I2+IP0                                        FIX  42
000063          I1MAX=I2+IP1-IP0                                    FIX  43
000065          DO 40 I1=I1MIN,I1MAX,IP0                            FIX  44
000067          DATA(I1)=DATA(J1)                                   FIX  45
000072          DATA(I1+1)=DATA(J1+1)                               FIX  46
000073    40    J1=J1+IP0                                           FIX  47
000077    50    DATA(I2+1)=DATA(J1)                                 FIX  48
000102    60    J1=J1+IP0                                           FIX  49
000106    70    DO 80 I2=1,IP2,IP1                                  FIX  50
000110          TEMPR=DATA(I2)                                      FIX  51
000112          DATA(I2)=DATA(I2)+DATA(I2+1)                        FIX  52
000114    80    DATA(I2+1)=TEMPR-DATA(I2+1)                         FIX  53
000120          IF (N-2) 200,200,90                                 FIX  54
000122    90    THETA=TWOPI/FLOAT(N)                                FIX  55
000136          SINTH=SIN(THETA/2.)                                 FIX  56
000153          ZSTPR=-2.*SINTH*SINTH                               FIX  57
000163          ZSTPI=SIN(THETA)                                    FIX  58
000167          ZR=(1.-ZSTPI)/2.                                    FIX  59
000210          ZI=(1.+ZSTPR)/2.                                    FIX  60
000231          IF (IFORM) 100,110,110                              FIX  61
000233   100    ZR=1.-ZR                                            FIX  62
000241          ZI=-ZI                                              FIX  63
000243   110    I1MIN=IP0+1                                         FIX  64
000245          I1MAX=IP0*(N/4)+1                                   FIX  65
000250          DO 190 I1=I1MIN,I1MAX,IP0                           FIX  66
000252          DO 180 I2=I1,IP2,IP1                                FIX  67
000253          I2CNJ=I2+IP0*(N/2)-2*(I1-1)                         FIX  68
000260          IF (I2-I2CNJ) 150,120,120                           FIX  69
000263   120    IF (ISIGN*(2*IFORM+1)) 130,140,140                  FIX  70
000267   130    DATA(I2+1)=-DATA(I2+1)                              FIX  71
000271   140    IF (IFORM) 170,180,180                              FIX  72
000273   150    DIFR=DATA(I2)-DATA(I2CNJ)                           FIX  73
000277          DIFI=DATA(I2+1)+DATA(I2CNJ+1)                       FIX  74
```

```
000301          TEMPR=DIFR*ZR-DIFI*ZI                              FIX  75
000316          TEMPI=DIFR*ZI+DIFI*ZR                              FIX  76
000332          DATA(I2)=DATA(I2)-TEMPR                            FIX  77
000335          DATA(I2+1)=DATA(I2+1)-TEMPI                        FIX  78
000337          DATA(I2CNJ)=DATA(I2CNJ)+TEMPR                      FIX  79
000342          DATA(I2CNJ+1)=DATA(I2CNJ+1)-TEMPI                  FIX  80
000343          IF (IFORM) 160,180,180                             FIX  81
000344     160  DATA(I2CNJ)=DATA(I2CNJ)+DATA(I2CNJ)                FIX  82
000347          DATA(I2CNJ+1)=DATA(I2CNJ+1)+DATA(I2CNJ+1)          FIX  83
000350     170  DATA(I2)=DATA(I2)+DATA(I2)                         FIX  84
000353          DATA(I2+1)=DATA(I2+1)+DATA(I2+1)                   FIX  85
000354     180  CONTINUE                                           FIX  86
000357          TEMP=ZR-.5                                         FIX  87
000365          ZR=ZSTPR*TEMP-ZSTPI*ZI+ZR                          FIX  88
000406     190  ZI=ZSTPR*ZI+ZSTPI*TEMP+ZI                          FIX  89
000431     200  IF (IFORM) 270,210,210                             FIX  90
              C UNPACK THE REAL TRANSFORM VALUES (TWO PER COLUMN)  FIX  91
000433     210  I2=IP2+1                                           FIX  92
000435          I1=I2                                              FIX  93
000436          J1=IPO*(N/2+1)*NR1 +1                              FIX  94
000443          GO TO 250                                          FIX  95
000443     220  DATA(J1)=DATA(I1)                                  FIX  96
000447          DATA(J1+1)=DATA(I1+1)                              FIX  97
000450          I1=I1-IPO                                          FIX  98
000451          J1=J1-IPO                                          FIX  99
000452     230  IF (I2-I1) 220,240,240                             FIX 100
000455     240  DATA(J1)=DATA(I1)                                  FIX 101
000461          DATA(J1+1)=0.                                      FIX 102
000462     250  I2=I2-IP1                                          FIX 103
000464          J1=J1-IPO                                          FIX 104
000466          DATA(J1)=DATA(I2+1)                                FIX 105
000470          DATA(J1+1)=0.                                      FIX 106
000471          I1=I1-IPO                                          FIX 107
000472          J1=J1-IPO                                          FIX 108
000472          IF (I2-1) 260,260,230                              FIX 109
000475     260  DATA(2)=0.                                         FIX 110
000476     270  RETURN                                             FIX 111
000477          END                                                FIX 112-
```