ASCII CODE CHART

This is a chart showing the decimal, hex, and ASCII codes available in the 8 bit ASCII standard code. Note that many codes, including null, carriage return, escape, and other reserved characters may not be entered directly, either because they are editing commands, or used to denote special functions. In this case, the code may be sent by a forward slash "/" followed by the decimal value of the code to enter into the macro string.

ASC	DEC	HEX	ASC	DEC	HEX	ASC	DEC	HEX	ASC	DEC	HEX	ASC	DEC	HEX	ASC	DEC	HEX
⊙	0	00	,	44	2C	X	88	58	ä	132	84	▒	176	B0	▄	220	DC
☺	1	01	-	45	2D	Y	89	59	à	133	85	▒	177	B1	▌	221	DD
☻	2	02	.	46	2E	Z	90	5A	å	134	86	▓	178	B2	▐	222	DE
♥	3	03	/	47	2F	[	91	5B	ç	135	87	│	179	B3	▀	223	DF
♦	4	04	0	48	30	\	92	5C	ê	136	88	┤	180	B4	α	224	E0
♣	5	05	1	49	31	]	93	5D	ë	137	89	╡	181	B5	ß	225	E1
♠	6	06	2	50	32	^	94	5E	è	138	8A	╢	182	B6	Γ	226	E2
●	7	07	3	51	33	_	95	5F	ï	139	8B	╖	183	B7	π	227	E3
◘	8	08	4	52	34	`	96	60	î	140	8C	╕	184	B8	Σ	228	E4
○	9	09	5	53	35	a	97	61	ì	141	8D	╣	185	B9	σ	229	E5
◙	10	0A	6	54	36	b	98	62	Ä	142	8E	║	186	BA	µ	230	E6
♂	11	0B	7	55	37	c	99	63	Å	143	8F	╗	187	BB	τ	231	E7
♀	12	0C	8	56	38	d	100	64	É	144	90	╝	188	BC	Φ	232	E8
♪	13	0D	9	57	39	e	101	65	æ	145	91	╜	189	BD	Θ	233	E9
♫	14	0E	:	58	3A	f	102	66	Æ	146	92	╛	190	BE	Ω	234	EA
☼	15	0F	;	59	3B	g	103	67	ô	147	93	┐	191	BF	δ	235	EB
►	16	10	<	60	3C	h	104	68	ö	148	94	└	192	C0	∞	236	EC
◄	17	11	=	61	3D	i	105	69	ò	149	95	┴	193	C1	φ	237	ED
↕	18	12	>	62	3E	j	106	6A	û	150	96	┬	194	C2	ε	238	EE
‼	19	13	?	63	3F	k	107	6B	ù	151	97	├	195	C3	∩	239	EF
¶	20	14	@	64	40	l	108	6C	ÿ	152	98	─	196	C4	≡	240	F0
§	21	15	A	65	41	m	109	6D	Ö	153	99	┼	197	C5	±	241	F1
▬	22	16	B	66	42	n	110	6E	Ü	154	9A	╞	198	C6	≥	242	F2
↨	23	17	C	67	43	o	111	6F	¢	155	9B	╟	199	C7	≤	243	F3
↑	24	18	D	68	44	p	112	70	£	156	9C	╚	200	C8	⌠	244	F4
↓	25	19	E	69	45	q	113	71	¥	157	9D	╔	201	C9	⌡	245	F5
→	26	1A	F	70	46	r	114	72	₧	158	9E	╩	202	CA	÷	246	F6
←	27	1B	G	71	47	s	115	73	ƒ	159	9F	╦	203	CB	≈	247	F7
∟	28	1C	H	72	48	t	116	74	á	160	A0	╠	204	CC	°	248	F8
↔	29	1D	I	73	49	u	117	75	í	161	A1	═	205	CD	∙	249	F9
▲	30	1E	J	74	4A	v	118	76	ó	162	A2	╬	206	CE	·	250	FA
▼	31	1F	K	75	4B	w	119	77	ú	163	A3	╧	207	CF	√	251	FB
	32	20	L	76	4C	x	120	78	ñ	164	A4	╨	208	D0	ⁿ	252	FC
!	33	21	M	77	4D	y	121	79	Ñ	165	A5	╤	209	D1	²	253	FD
"	34	22	N	78	4E	z	122	7A	ª	166	A6	╥	210	D2	■	254	FE
#	35	23	O	79	4F	{	123	7B	º	167	A7	╙	211	D3		255	FF
$	36	24	P	80	50	|	124	7C	¿	168	A8	╘	212	D4			
%	37	25	Q	81	51	}	125	7D	⌐	169	A9	╒	213	D5			
&	38	26	R	82	52	~	126	7E	¬	170	AA	╓	214	D6			
'	39	27	S	83	53	⌂	127	7F	½	171	AB	╫	215	D7			
(	40	28	T	84	54	Ç	128	80	¼	172	AC	╪	216	D8			
)	41	29	U	85	55	ü	129	81	¡	173	AD	┘	217	D9			
*	42	2A	V	86	56	é	130	82	«	174	AE	┌	218	DA			
+	43	2B	W	87	57	â	131	83	»	175	AF	▉	219	DB			

SYNTHESIS EXTENDED MACRO STRING MAPPING

It is possible to extend the effective number of ASCII string messages that may be sent by limiting the number of characters per macro to 16. In this case, each macro line is divided into two 16 byte strings, which may be accessed as follows:

MACRO	LO16	HI16	MACRO	LO16	HI16
1	129	130	33	193	194
2	131	132	34	195	196
3	133	134	35	197	198
4	135	136	36	199	200
5	137	138	37	201	202
6	139	140	38	203	204
7	141	142	39	205	206
8	143	144	40	207	208
9	145	146	41	209	210
10	147	148	42	211	212
11	149	150	43	213	214
12	151	152	44	215	216
13	153	154	45	217	218
14	155	156	46	219	220
15	157	158	47	221	222
16	159	160	48	223	224
17	161	162	49	225	226
18	163	164	50	227	228
19	165	166	51	229	230
20	167	168	52	231	232
21	169	170	53	233	234
22	171	172	54	235	236
23	173	174	55	237	238
24	175	176	56*	239	240
25	177	178	57	241	242
26	179	180	58	243	244
27	181	182	59	245	246
28	183	184	60	247	248
29	185	186	61	249	250
30	187	188	62	251	252
31	189	190	63**	253	254
32	191	192	64		

\ terminates string (remainder is comment)
| send ESCAPE character
/nn Send any ASCII character nn (00-$FF)
/nnX Send ASCII character = X(nn)
/nnA Send ASCII character from analog channel (nn)
/C Calculate and send a checksum of all bytes in sent in this macro

* last available for TC-3550
** last accessible on Synthesis

PIONEER 8000 CODE CHART

CONTROL COMMANDS
RxD Terminal Control Command Input

Explanation of commands
Below is the list of the commands that can be carried out on the LD-V8000.

	COMMAND	MNEMONIC		COMMAND	MNEMONIC
1	Door Open	OP	36	Reg. D Set (RS-232C)	argument RD
2	Reject	RJ	37	Reg. E Set (Use Switch)	argument RE
3	Start	SA	38	Reg. F Set (Use Switch)	argument RF
4	Play	(address) PL	39	Reg. G Set (Video Mem.)	argument RG
5	Pause	PA	40	Reg. H Set (Extended)	argument RH
6	Still	ST	41	Clear Screen	CS
7	Step Forward	SF	42	Print	argument PR
8	Step Reverse	SR	43	Lead Out Symbol	LO
9	Scan Forward	NF	44	Multi Track Jump Forward	argument JF
10	Scan Reverse	NR	45	Multi Track Jump Reverse	argument JR
11	Multi-speed Forward	(address) MF	46	User's Code Request	?U
12	Multi-speed Reverse	(address) MR	47	Sub Audio Control	argument AS
13	Speed	argument SP	48	Set Video Memory Mode	argument MM
14	Search	address SE	49	Video Memory	argument VM
15	Stop Marker	address SM	50	Enable Video Memory Cntl	argument EM
16	Frame	FR	51	Disable Video Memory Cntl	argument DM
17	Time	TM	52	Reg. A Request (Display)	$A
18	Chapter	CH	53	Reg. B Request (SQ Cont)	$B
19	Audio Control	argument AD	54	Reg. C Request (Misc.)	$C
20	Video Control	argument VD	55	Reg. D Request (RS-232C)	$D
21	Display Control	argument DS	56	Reg. E Request (User Swt)	$E
22	Clear	CL	57	Reg. F Request (User Swt)	$F
23	Frame Number Request	?F	58	Reg. G Request (Vid Mem)	$G
24	Time Code Request	?T	59	Reg. H Request (Extend)	$H
25	Chapter Number Request	?C	60	Select Input Unit	argument #S
26	Player Active Mode Reqst	?P	61	Input Unit Request	#I
27	Disc Status Request	?D	62	Input Number Wait	?N
28	Communication Control	argument CM	63	Sex Aux. Port	argument #A
29	CCR Mode Request	?M	64	Beep Control	argument BP
30	Door Close	CO	65	Set Program Pointer	argument *S
31	LVP Model Name Request	?X	66	Program Read	(argument) *D
32	Key Lock	argument KL	67	Program Write	(argument) *W
33	Reg. A Set (Display)	argument RA	68	Program Counter Request	*P
34	Reg. B Set (SQ Cont)	argument RG	69	Program Run	argument *R
35	Reg. C Set (Miscellany)	argument RC	70	Program Halt	*H

- The commands No. 1 through No. 29 are general commands which are called Common Commands.
- Command mnemonics use ASCII alpha characters and the ? symbol. There is no distinction between capitals and small letters.
- The argument and address are decimal and use ASCII numerals.
- An address indicated by ( ) can be omitted.
- A command line can have up to 20 characters and is terminated with CR code (ODH).
- Refer to the "Users Manual" for the control protocol and command explanation.

AUTOMATIC CONTROL COMMANDS

COMMAND	MNEMONIC	HEX	COMMAND	MNEMONIC	HEX
Mode Control Commands			Automatic Stop	(address) AS	F3
Halt Program Control	H	BF	Step Frame Mode	SFM	8E
Pgm Load Cntl Command			Set Time Mode	STM	8D
Set Active Memory Page	(address) PAG	11	Set Chapter Mode	SCM	8C
Load Program from Disc	L	CC	Slow Speed Set	argument SS	ED
Move Load	argument L	CC	Fast Speed Set	argument FS	EC
Partial Load	PLD	0C	Set 4 Field Still Mode	(argument) SSM	8B
Move Partial Load	argument PLD	0C	Register Mgmt Commands		
Audio Control Commands			Add to Register 0	argument ADD	02
CX Control	(argument) CX	EE	Subtract # from Register 0	argument SUB	03
Audio Output Control	(argument) A1	F4	Multiplication	argument MUL	22
Audio Output Control	(argument) A2	FC	Division	argument DIV	21
Set Audio Status	AFF	A0	Argument	(argument) ARG	0A
Set Audio Status	AFN	A1	Compare Contents Reg. 0	argument COM	04
Set Audio Status	ANF	A2	Decrement Register	argument DR	F0
Set Audio Status	ANN	A3	Drop Low-Order Digit	DRP	1D
Set Audio Status	AFT	A4	Set Value into Reg. 0	argument GET	08
Set Audio Status	AFI	A5	Transfer Value from Reg. 0	argument PUT	09
Set Audio Status	ANT	A6	Recall an Active Register	(argument) RC	7F
Set Audio Status	ANI	A7	Generate “Random” #	RND	05
Set Audio Status	ATF	A8	Store # in Active Reg.	(argument) ST	F5
Set Audio Status	ATN	A9	Read Rear Switch	RRS	10
Set Audio Status	AIF	AA	Clock Read and Reset	CLK	16
Set Audio Status	AIN	AB	Input Processing Cmds		
Set Audio Status	ATT	AC	Input View Response	argument IN	F8
Set Audio Status	ATI	AD	View Input w/Function Keys	argument FIN	18
Set Audio Status	AIT	AE	View Input w/Timeout	argument TIN	0E
Digital Audio Output	(argument) DAD	82	View Input w/F.K and Time	argument FTI	19
Video Control Commands			Data Input	DIN	1E
Video Off	VFF	1C	Binary Input	BIN	17
Video On	VON	1B	Interrupt Input	(argument) IIN	20
Display Control Command			Pgm Execution Cntl Cmds		
Frame Display	(argument) DI	F1	Transfer to Location	(argument) BR	CF
Set User Display	argument SUD	2B	Brand on Failure	(argument) BRF	07
Clear Display	(argument) CLD	2C	Jump to Subroutine	(argument) JMP	0B
Blink	(argument) BLK	2D	No Entry	NE	FF
Clear Blink	(argument) CLB	2R	Flag Set Commands		
Set Background Color	(argument) SBC	88	Character Gen. Enable	CGE	E0
Player Control Commands			Character Gen. Disable	CGD	E1
Reject Videodisc	RJ	F9	RCU Enable	RCE	12
Pause in Execution	argument PAU	0D	RCU Disable	RCD	13
Play Motion Seq @ Normal	(address) P	FD	Set Player Control Switch	argument SCS	8A
Search to a Frame	(address) SC	F7	Transmit Commands		
Multi-speed Play Forward	(address) MF	F2	Set Transmit Pointer	argument STP	E8
Multi-speed Play Reverse	(address) MR	FA	Increment Transmit Pointer	ITP	D9
Freeze Frame	W	FB	Decrement Transmit Pointer	DTP	DA
Freeze Frame and Delay	argument W	FB	Transmit Memory (RAM)	TM	DC
Step Single Frame Fwd	SF	F6	Increment Transmit Mem	ITM	DD
Set Single Frame Reverse	SR	FE	Decrement Transmit Mem	DTM	DE

SONY LDP COMMAND LISTS

SONY LDP-1500 COMMAND LIST

	0	1	2	3	4	5	6	7
0				0	Enter	Index On	Addr Inq	
1	Completion			1	C.E.	Index Off	Continue	Non C.F. Play
2	Error			2	Menu		Motor On	ROM Version Inq
3	Lid Open			3	Search		Motor Off	Mark Set
4			Audio Mute On	4	Repeat			Eject Enable
5	Not Target		Audio Mute Off	5		Frame # Mode		Eject Disable
6	No Frame #		Video Off	6	Ch-1 On	C.L.		Chapter # Inq
7	Mark Return		Video On	7	Ch-1 Off		Status Inq	
8			PSC Enable	8	Ch-2 On			
9			PSC Disable	9	Ch-2 Off		Chapter # Mode	
A	Ack		Eject	F-Play	R-Play	Memory		
B	Nak		Fwd Step & Still	F-Fast	R-Fast	M-Search		
C			Rev Step & Still	F-Slow	R-Slow			
D				F-Step	R-Step			
E				F-Scan	R-Scan			
F				Stop	Still			

SONY LDP-2000-1 COMMAND LIST

	0	1	2	3	4	5	6	7
0				0	@ Enter	P Index On	` Addr Inq	p CX
1	Completion			1	A C.E.	Q Index Off	a Continue	q Non C.F. Play
2				2			b Motor On	r ROM Version Inq
3				3	C Search		c Motor Off	
4				4	D Repeat		d Ch-1	
5	Not Target			5		U Frame # Mode	e Ch-2	
6	No Frame #			6	F Ch-1 On	V C.L.	f Index	
7				7	G Ch-1 Off		g Status Inq	
8				8	H Ch-2 On			
9				9	I Ch-2 Off		I Chapter # Mode	
A	Ack			: F-Play	J R-Play	Z Memory		
B	Nak			; F-Fast	K R-Fast	[M-Search		
C				< F-Slow	L R-Slow			
D				= F-Step	M R-Step			
E				> F-Scan	N R-Scan		n CX On	
F				? Stop	O Still	- Mode	o CX Off	

AUTOPATCH 1YDM SERIAL COMMUNICATION PROTOCOL

(also known as Soundelux MR-16)

KEYS	FUNCTION	DESCRIPTION
X	Cancel	Cancels previous incomplete string
T	Take	Executes command string
S	Status	Places DM into Status Mode
L	Matrix	Allows a matrix to be selected
C	Change	Places DM into Change Mode
I	Input	Input number entry
O	Output	Output number entry
R	Recall	Executes preset configuration
RR	Store	Defines preset configuration
-	Space	A Delimiter for separating multiple inputs or outputs
0-9	Number	Digits used to assemble input, output, and preset numbers
P	Program	Places DM into Program Mode
D	Disconnect	Disconnects an input or output

EXAMPLES	DESCRIPTION
CI3O4T	Change input 3 to Output 4 on all matrices. (The command works on all matrices because no matrix was specified).
CO4 5 6 8I1T	Change Outputs 4,5,6 and 8 on all matrices to receive input 1. The order in which Inputs and outputs are specified in a command does not matter. When specifying multiple Outputs, put a space between the signal numbers.
SL4I2T	Check the Status of Input 2 on Matrix 4.
CL1I4XCL2I4O5T	Change, on Matrix 1, input 4 was canceled. Change Input 4 to Output 5 on Matrix 2.
RR3T	Define the current I/O configuration as Preset 3.
R2T	Recall preset 2.

SERIAL COMMUNICATIONS WITH THE AUTOPATCH

IBM PC: DB9	AutoPatch: DB9
RD #2	TD #3
TD #3	RD #2
SG #5	SG #5
DSR #6	DTR #4
RTS #7	CTS #8

232 BART: DB9	AutoPatch: DB9
RD #8	TD #3
TD #4	RD #2
SG #5	SG #5

TECHTERM OPERATION

Triad has tested and used various models of TechTerm terminals from Two Technologies, Inc. and have found the following function key definitions to help with common operations with the diagnostics and focus remote subsystems. Across the top row of keys, there are five function keys, labeled F1 to F5. These keys may also be used in conjunction with the SHIFT and CTRL keys to create additional shortcuts. Any other compatible terminal or emulator may be used.

Our standard definition is:

F1	F2	F3	F4	F5
ESC	/	-	<,	.>
	(next)	(to)	(lower)	(raise)

To re-program a function key, press CTRL-SHIFT-F2. The terminal will prompt for the function key to define. Press the function key to define, F1 through F5. (These may be combined with the CTRL and SHIFT keys for additional functions.) The current key definition in **hex** will be shown, along with the option to Change [F1], Next[F2], or Exit [F5]. Press F1 to define the new code, from the partial table below, then press F5 to save the code in memory.

KEY	CHARACTER	HEX CODE
F1	ESCAPE	1B
F2	/	2F
F3	-	2D
F4	, (lower)	2C
F5	. (raise)	2E

STANDARD SETUP

The operating parameters are stored in the TechTerm's non-volatile memory, and are accessed by pressing CTRL-SHIFT-F1. The configuration we have used is:

BAUD = 9600
DATA BITS = 7
PARITY = IGNORE
DISPLAY PE DISABLED
REPEAT = SLOW
ECHO = DISABLED
HANDSHAKE = DISABLED
SELF TEST ENABLED (OPTIONAL!)

TERMINAL DATA CABLE

A special modular cable is needed between the Triad TC-560 Transmitter card and the TechTerm terminal, and it is important that the correct end be used at the terminal, as the 5 volt power for the terminal is supplied through the cable. Under NO CONDITION attempt to plug any of the modular connectors to a telephone line, as the 48 volt/110 volt ring signal **MAY** DESTROY the terminal or processor card!

6 PIN MODULAR RS-232 TERMINAL CONNECTOR				**TECHTERM RS-232 CONNECTOR**		
PIN	**DESCRIPTION**	**COLOR**		**PIN**	**DESCRIPTION**	**COLOR**
1	No Connection	Blue	----------	3	HSK Out	Blue
2	Common	Yellow	----------	2	HSK In	Yellow
3	RXD	Green	----------	5	TXD	Green
4	TXD	Red	----------	4	RXD	Red
5	Common	Black	----------	6	Common	Black
6	+5VDC	White	----------	1	+5VDC	White

LIGHTING CHANNELS TO DIGITAL CHANNELS/SUBCHANNELS CHART

LGT	CH	SUB	LGT	CH	SUB	LGT	CH	SUB	LGT	CH	SUB
1	1	1	65	3	1	129	5	1	193	7	1
2	1	2	66	3	2	130	5	2	194	7	2
3	1	3	67	3	3	131	5	3	195	7	3
4	1	4	68	3	4	132	5	4	196	7	4
5	1	5	69	3	5	133	5	5	197	7	5
6	1	6	70	3	6	134	5	6	198	7	6
7	1	7	71	3	7	135	5	7	199	7	7
8	1	8	72	3	8	136	5	8	200	7	8
9	1	9	73	3	9	137	5	9	201	7	9
10	1	10	74	3	10	138	5	10	202	7	10
11	1	11	75	3	11	139	5	11	203	7	11
12	1	12	76	3	12	140	5	12	204	7	12
13	1	13	77	3	13	141	5	13	205	7	13
14	1	14	78	3	14	142	5	14	206	7	14
15	1	15	79	3	15	143	5	15	207	7	15
16	1	16	80	3	16	144	5	16	208	7	16
17	1	17	81	3	17	145	5	17	209	7	17
18	1	18	82	3	18	146	5	18	210	7	18
19	1	19	83	3	19	147	5	19	211	7	19
20	1	20	84	3	20	148	5	20	212	7	20
21	1	21	85	3	21	149	5	21	213	7	21
22	1	22	86	3	22	150	5	22	214	7	22
23	1	23	87	3	23	151	5	23	215	7	23
24	1	24	88	3	24	152	5	24	216	7	24
25	1	25	89	3	25	153	5	25	217	7	25
26	1	26	90	3	26	154	5	26	218	7	26
27	1	27	91	3	27	155	5	27	219	7	27
28	1	28	92	3	28	156	5	28	220	7	28
29	1	29	93	3	29	157	5	29	221	7	29
30	1	30	94	3	30	158	5	30	222	7	30
31	1	31	95	3	31	159	5	31	223	7	31
32	1	32	96	3	32	160	5	32	224	7	32
33	2	1	97	4	1	161	6	1	225	8	1
34	2	2	98	4	2	162	6	2	226	8	2
35	2	3	99	4	3	163	6	3	227	8	3
36	2	4	100	4	4	164	6	4	228	8	4
37	2	5	101	4	5	165	6	5	229	8	5
38	2	6	102	4	6	166	6	6	230	8	6
39	2	7	103	4	7	167	6	7	231	8	7
40	2	8	104	4	8	168	6	8	232	8	8
41	2	9	105	4	9	169	6	9	233	8	9
42	2	10	106	4	10	170	6	10	234	8	10
43	2	11	107	4	11	171	6	11	235	8	11
44	2	12	108	4	12	172	6	12	236	8	12
45	2	13	109	4	13	173	6	13	237	8	13
46	2	14	110	4	14	174	6	14	238	8	14
47	2	15	111	4	15	175	6	15	239	8	15
48	2	16	112	4	16	176	6	16	240	8	16
49	2	17	113	4	17	177	6	17	241	8	17
50	2	18	114	4	18	178	6	18	242	8	18
51	2	19	115	4	19	179	6	19	243	8	19
52	2	20	116	4	20	180	6	20	244	8	20
53	2	21	117	4	21	181	6	21	245	8	21
54	2	22	118	4	22	182	6	22	246	8	22
55	2	23	119	4	23	183	6	23	247	8	23
56	2	24	120	4	24	184	6	24	248	8	24
57	2	25	121	4	25	185	6	25	249	8	25
58	2	26	122	4	26	186	6	26	250	8	26
59	2	27	123	4	27	187	6	27	251	8	27
60	2	28	124	4	28	188	6	28	252	8	28
61	2	29	125	4	29	189	6	29	253	8	29
62	2	30	126	4	30	190	6	30	254	8	30
63	2	31	127	4	31	191	6	31	255	8	31
64	2	32	128	4	32	192	6	32	256	8	32

LGT	CH	SUB	LGT	CH	SUB	LGT	CH	SUB	LGT	CH	SUB
257	9	1	321	11	1	385	13	1	449	15	1
258	9	2	322	11	2	386	13	2	450	15	2
259	9	3	323	11	3	387	13	3	451	15	3
260	9	4	324	11	4	388	13	4	452	15	4
261	9	5	325	11	5	389	13	5	453	15	5
262	9	6	326	11	6	390	13	6	454	15	6
263	9	7	327	11	7	391	13	7	455	15	7
264	9	8	328	11	8	392	13	8	456	15	8
265	9	9	329	11	9	393	13	9	457	15	9
266	9	10	330	11	10	394	13	10	458	15	10
267	9	11	331	11	11	395	13	11	459	15	11
268	9	12	332	11	12	396	13	12	460	15	12
269	9	13	333	11	13	397	13	13	461	15	13
270	9	14	334	11	14	398	13	14	462	15	14
271	9	15	335	11	15	399	13	15	463	15	15
272	9	16	336	11	16	400	13	16	464	15	16
273	9	17	337	11	17	401	13	17	465	15	17
274	9	18	338	11	18	402	13	18	466	15	18
275	9	19	339	11	19	403	13	19	467	15	19
276	9	20	340	11	20	404	13	20	468	15	20
277	9	21	341	11	21	405	13	21	469	15	21
278	9	22	342	11	22	406	13	22	470	15	22
279	9	23	343	11	23	407	13	23	471	15	23
280	9	24	344	11	24	408	13	24	472	15	24
281	9	25	345	11	25	409	13	25	473	15	25
282	9	26	346	11	26	410	13	26	474	15	26
283	9	27	347	11	27	411	13	27	475	15	27
284	9	28	348	11	28	412	13	28	476	15	28
285	9	29	349	11	29	413	13	29	477	15	29
286	9	30	350	11	30	414	13	30	478	15	30
287	9	31	351	11	31	415	13	31	479	15	31
288	9	32	352	11	32	416	13	32	480	15	32
289	10	1	353	12	1	417	14	1	481	16	1
290	10	2	354	12	2	418	14	2	482	16	2
291	10	3	355	12	3	419	14	3	483	16	3
292	10	4	356	12	4	420	14	4	484	16	4
293	10	5	357	12	5	421	14	5	485	16	5
294	10	6	358	12	6	422	14	6	486	16	6
295	10	7	359	12	7	423	14	7	487	16	7
296	10	8	360	12	8	424	14	8	488	16	8
297	10	9	361	12	9	425	14	9	489	16	9
298	10	10	362	12	10	426	14	10	490	16	10
299	10	11	363	12	11	427	14	11	491	16	11
300	10	12	364	12	12	428	14	12	492	16	12
301	10	13	365	12	13	429	14	13	493	16	13
302	10	14	366	12	14	430	14	14	494	16	14
303	10	15	367	12	15	431	14	15	495	16	15
304	10	16	368	12	16	432	14	16	496	16	16
305	10	17	369	12	17	433	14	17	497	16	17
306	10	18	370	12	18	434	14	18	498	16	18
307	10	19	371	12	19	435	14	19	499	16	19
308	10	20	372	12	20	436	14	20	500	16	20
309	10	21	373	12	21	437	14	21	501	16	21
310	10	22	374	12	22	438	14	22	502	16	22
311	10	23	375	12	23	439	14	23	503	16	23
312	10	24	376	12	24	440	14	24	504	16	24
313	10	25	377	12	25	441	14	25	505	16	25
314	10	26	378	12	26	442	14	26	506	16	26
315	10	27	379	12	27	443	14	27	507	16	27
316	10	28	380	12	28	444	14	28	508	16	28
317	10	29	381	12	29	445	14	29	509	16	29
318	10	30	382	12	30	446	14	30	510	16	30
319	10	31	383	12	31	447	14	31	511	16	31
320	10	32	384	12	32	448	14	32	512	16	32

DISPLAYING MESSAGES ON SYNTHESIS MONITOR (COLOR AND POSITION)

The following table shows the color and position of text displays on the Synthesis monitor. This is accessed by using Event type 16 (Disply) under CPU Events. Disply (event type 16) displays the text of the specified macro number in PARM2, based on the color and position information in PARM1. For example, if you wanted text macro number 3 to be displayed on the fifth line in Red, you would enter:

```
00:10.00  Disply 068 003
```

	Black	Blue	Green	Cyan	Red	Purple	Brown	Grey
Line 1	0	16	32	48	64	80	96	112
Line 2	1	17	33	49	65	81	97	113
Line 3	2	18	34	50	66	82	98	114
Line 4	3	19	35	51	67	83	99	115
Line 5	4	20	36	52	68	84	100	116
Line 6	5	21	37	53	69	85	101	117
Line 7	6	22	38	54	70	86	102	118
Line 8	7	23	39	55	71	87	103	119
Line 9	8	24	40	56	72	88	104	120
Line 10	9	25	41	57	73	89	105	121
Line 11	10	26	42	58	74	90	106	122
Line 12	11	27	43	59	75	91	107	123
Line 13	12	28	44	60	76	92	108	124
Line 14	13	29	45	61	77	93	109	125
Line 15	14	30	46	62	78	94	110	126
Line 16	15	31	47	63	79	95	111	127

	F	L	A	S	H	I	N	G
Line 1	128	144	160	176	192	208	224	240
Line 2	129	145	161	177	193	209	225	241
Line 3	130	146	162	178	194	210	226	242
Line 4	131	147	163	179	195	211	227	243
Line 5	132	148	164	180	196	212	228	244
Line 6	133	149	165	181	197	213	229	245
Line 7	134	150	166	182	198	214	230	246
Line 8	135	151	167	183	199	215	231	247
Line 9	136	152	168	184	200	216	232	248
Line 10	137	153	169	185	201	217	233	249
Line 11	138	154	170	186	202	218	234	250
Line 12	139	155	171	187	203	219	235	251
Line 13	140	156	172	188	204	220	236	252
Line 14	141	157	173	189	205	221	237	253
Line 15	142	158	174	190	206	222	238	254
Line 16	143	159	175	191	207	223	239	255

SCU MEMORY MAP

Under this release of the SCU/LDC/BART software, the events data are located in the hex range $2000 - $3FFF on the TC-3550 CPU board.

ADDRESS	DESCRIPTION
$0017	Software DIP Switch
	$00=No Output, $10=Running Display, $40=SMPTE
$0018 - $001F	Hi byte of highest address in TC-3505 Memory Expander - $20* for 2764, $40 for 27128, $80 for 27256 for ROMS 1-8.
$002E	Cold Start
$00D4	# of Analog Channels
$00D6	# of Digital Channels
$00FB	SMPTE Hours Field
$0280	64: Analog channel configuration bytes
(96.10 & Higher)	$00=Off, $10=8 Mono polar, $20=Uninitialized, $40=8 Bipolar, $80=12 Bipolar
$02C0	Preset Array
$0700 - $074F	Analog values for all 64 analog channels
$0A40 - $0A4F	Macro Status
$0D40 - $0D5F	Trigger Memory
$0F00 - $0FFF	Realtime Scheduler Table (only if using Battery-backed Real time Clock)
$2000 - $2FFF	Events
$3002 - $35FF	Subroutine search starts at $2000 and must be divisible by 6. It is set based on the division between timed and subroutine events.
	Revision 96.04 and earlier used $3600 - $37EF for ON Events
$3600 - $37FF	ON:Events consisting of 16 3-byte sequences
$3600 - $362F	16: 3 byte ON RESET events
$3630 - $365F	16: 3 byte ON START events
$3660 - $368F	16: 3 byte ON ABORT events
$3690 - $36BF	16: 3 byte ON END events
$36C0 - $36EF	16: 3 byte ON E.O.T. events
	Revision 96.10 and later use $3700 - $37EF for ON Events
$3700 - $37EF	ON:Events consisting of 16 3-byte sequences
$3700 - $372F	16: 3 byte ON RESET events
$3730 - $375F	16: 3 byte ON START events
$3760 - $378F	16: 3 byte ON ABORT events
$3790 - $37BF	16: 3 byte ON END events
$37C0 - $37EF	16: 3 byte ON E.O.T. events
$3800 - $3EFF $3800 - $3F3F	56: 32 byte macro strings (or 112, 16 byte strings using the extended addressing codes) **(62 in rev 96.10)**
$3F00 - $3F3F	64: Analog channel configuration bytes
(96.04 & lower)	$00=Off, $10=8 Mono polar, $20=Uninitialized, $40=8 Bipolar, $80=12 Bipolar
**$3F00 - $3F3F	32:2 byte (low/hi) cue subroutine offsets
**$3F40 - $3F5F	32:1 byte bank ID, 0=standard memory, 1-8=TC-3505 bank
$3FC0 - $3FCF	Table of Com 1, 2, 3, 4 inits, 4 bytes each. These are loaded from this table into the appropriate UART registers.
**	Note: TC-3550 Rev. E and higher processors, and all TC-550 "BART" Processors use EXAR UART parameters for the first two (LDC) or four (BART) communications ports.

Example Using COM1:				
Address:	$3FC0	$3FC1	$3FC2	$3FC3
Value:	$BB	$13	$07	$00
Parameter	Baud Rate	Bit/Parity	Stop Bits	Reserved
Use:	9600	8 Bit, No Parity	1 Stop Bit	N/A
Baud Values: (C0, C4, C8, CC)	$66=1200 BPS, $88=2400 BPS, $99=4800 BPS, $BB=9600 BPS, $CC=38.4 KBPS, $EE=External (Used for DMX-512			
Data/Parity:	$13=8 bit, No parity, $			

ADDRESS	**DESCRIPTION**
$3FD0 - $3FEF	Table of TC-3518 COM inits, 4 bytes each. These currently are based on the Rockwell 65C52 UART parameters. A listing of these values is given in this section.
$3FF0	Frame rate, either $0F for 15 FPS or $1E for 30 FPS*
$3FF1	Default sync mode upon startup (may be overridden in ON RESET)
$3FF2	Low byte clock divider - $15 for 1 MHz, $2A for 2 MHz*
$3FF3	Hi byte clock divider - $41 for 1 MHz, $82 for 2 MHz*
$3FF4	Number of analog moves, $40 (64) is default/maximum*
$3FF5	Analog output polarity, EOR'd at output time ($FF*)
$3FF6	Number of digital subchannels, $10 for 16 channel, $18 for 24*
$3FF7	Digital output polarity, EOR'd at output ($00*)
$3FF8 - $3FFF	Hi byte of highest address in TC-3505 Memory Expander - $20* for 2764, $40 for 27128, $80 for 27256 for ROMS 1-8.

If the parm table at $3FF0 is not present, system defaults will be used. If present, as identified by a valid frame rate (15 or 30 FPS), then all subsequent values ($3FF0 - $3FFF) must be entered, and must be correct! Please confirm with Triad the correct values for the specific application if you intend to use this option.

SCU REAL TIME SCHEDULER

General

The Real Time Scheduler (RTS) module allows entry, editing, and performance of functions at predefined times during a weekly schedule. For each entry, it is possible to select the day or days, hours and minutes, and the function which is to occur. Beginning with release 88.07, up to 64 schedule entries are possible.

A battery backed, real-time clock (RTC) is added to the CPU (processor) card and is used to set the internal clock daily at midnight or whenever the system is restarted. Also at midnight, all events flagged as processed are cleared and the day is advanced.

NOTE:
- The clock module is installed in ROM socket U__ on a TC-3500 processor.
- The clock module is installed in ROM socket U11 on a TC-3550 processor card.
- The clock module is installed under the program ROM U12 in a Bart system.

The schedule information is normally kept in battery-backed RAM memory so that editing may be performed on site. It is also possible to store the schedule information in EPROM for permanent installations.

NOTE: The normal run-time display shows minutes, seconds, and frames relative to SHOW TIME (or SMPTE time code), and bears no relationship to the real time of day used for scheduling operations.

The system must be in the normal run (execution) mode for any scheduled events to occur. If the system is restarted, the system will catch up to real time, performing all unexecuted scheduled events up to the current time of day.

It is not necessary for the scheduled entries to be sorted, or entered in any particular order, other than for ease of maintenance and updating.

The scheduler is an optional feature and may not be present on all systems.

EDITING THE SCHEDULE

The schedule menu allows four options: displaying or editing the event schedule, displaying or changing the current day and time, clearing ALL schedule entries, or ESC.aping to the Main Menu.

```
TIME: E.dt S.et C.lr E.sc
```

Press [E] to enter the schedule editor. An event schedule entry appears on the display terminal as follows:

```
SMtWtFS 12:43*002 <04
```

The letters correspond to the day of the week, starting with Sunday as day 1. Any CAPITAL letters indicate that the function is to be performed on that day. Lower case letters indicate that the function should be skipped. The time display shows the hours and minutes that the event is to occur, in 24 hour military time format (00= 12:00 midnight, 12:00= 12:00 noon). An asterisk (*) after the time indicates that the function has occurred (since being cleared at midnight or reset). The function code that follows is the number of the EVENT SUBROUTINE to execute when the current real time matches or is greater than the scheduled time. The "<04" indicates the relative index of the schedule entry, from 01 to 64 (was 32).

When the entry is displayed, the cursor is positioned to the first day of the week (Sunday). The following keys will perform the indicated action:

[C] Will toggle the C.urrent day from active (upper case) to inactive (lower case), or vice versa. The cursor is advanced to the next day.

[SPACE] Will step to the next day without changing the status.

[B] Steps B.ack to the previous day without changing the status.

TAB Advances the cursor to the HOURS field.

ESCAPE Will set the cursor back to the start of the line, unless already at the first position. In this case, the schedule menu is shown.

[.>] Will advance the index and display the next scheduled event.

[,<] Will decrement the index and display the previous event.

NOTE: The CAPS LOCK must be on (as upper case is expected). Do NOT shift the comma/period key to step forward or back.

At the HOURS field, enter the hour of the day the event is to occur, from 00 to 23. Terminate with either a RETURN or the SPACE bar. Or use the TAB key to advance to the next field without changing the current HOURS.

At the MINUTES field, enter the minutes, from 00 to 59 and terminate with either the SPACE BAR or a RETURN. Or press TAB to advance to the next field without changing the current MINUTES. The FUNCTION code is a number from 01 to 250, and corresponds to the EVENT SUBROUTINE which is to be performed. A function code of 00 indicates that no function is to occur.

Pressing ESCAPE at any field returns the cursor to the start of the line.

SETTING THE CLOCK

Press S.et from the schedule menu. The day and time are shown as:

1 12:45:14

The first number is the day of the week, starting with Sunday as #1. The time is displayed and entered in 24 hour military time format, as hours (00-23), minutes (00-59), and seconds (00-59).

Press ESCAPE to terminate editing, otherwise enter all four values (separated by either the SPACE bar or a RETURN). If the clock module is installed, the time will be updated in the battery-backed clock.

Normally, the computer's time of day clock will be updated every night at midnight if power is left on; otherwise, the time is reset when the system is powered up.

FUNCTION CODES

The function code (00-255) actually calls an EVENT SUBROUTINE, which has been pre-programmed for the application.

Refer to the documentation for the specific installation to identify each of the possible pre-programmed function codes.

CLEARING THE SCHEDULE

Normally the schedule will be maintained in battery-backed RAM or made semi-permanent in EPROM. If no battery RAM is available, the schedule will be lost if power is lost. The schedule may be entirely zapped by pressing C.LR from the schedule menu, or by a system cold initialization. The computer will ask "Are you Sure?". Enter "Y" for yes, meaning you want to clear the schedule, or "N" for no, cancel this operation!

NOTE: There is now a DOS program that will allow you to convert a SYNTHESIS Schedule file (*.SCH) into an upload file, making the editing of the real-time schedule easier to do. Contact Triad for the program and information on using it.

NOTES AND UPDATES

02/15/88 First Release.
03/15/88 Reworked editing key codes.
07/21/88 Extended schedule entries to 64.

USE OF FADERT AND FADETO COMMANDS EXPLAINED FURTHER

FADERATE.TXT
03/20/92

The Events 34 and 35 (FadeRt and FadeTo) are designed to provide an automatic linear fade on any of the analog channels configured on the system. Both events are needed to start a fade on a given channel, as three total parameters are required (and the events system currently only supports two). The FadeRt must be placed before the FadeTo, and occur at the same time (in the same frame).

An example follows:

```
01:00:01  FadeRt 012 008  (event type 34) use channel 12 at rate 8
01:00:01  FadeTo 012 254  (event type 35) fade channel 12 to 254
```

The analog channel will begin stepping at a nominal RATE of "8" to the target value, 254. Note that the rate is related to an increment (or decrement) based on counts per frame (or frames per count) to generate a constant RATE for any given ramp. In the Synthesis Lighting system, this rate value is calculated automatically based on the current value, the target value, and the number of SECONDS requested. Further note that the rate value must be calculated based on the current system frame rate (15/24/30 FPS). The system will automatically determine the direction of the fade and stop when the target level is reached.

- Currently, the analog channel specified in the FadeRt (34) event is a "dummy" variable (useful for documentation), but this rate remains in effect for ANY other channels that are given a FadeTo (35) command. This allows multiple channels to ramp at the same rate without having to re-specify the FadeRt time.

Some rough timing for a ramp from 0 to 255 (or 255 to 0) follow.

	SECONDS	**SECONDS**
Rate Value	**@ 15 FPS**	**@ 30 FPS**
1	256	128
2	128	64
4	64	32
8	32	16
16	16	8
32	8	4
64	4	2
128	2	1
255	1	.5

(Note that any number may be used, and can be interpolated from the table above.) If a faster fade is required, either use the real-time system or give discrete levels as Events.

The longest fade previously available was 32 seconds (@ 15 FPS), using a rate of "1". This has been extended as shown in the table to provide a wider selection and longer fade times.

ABOUT SUBROUTINES, ARMEVT (ARMED EVENTS) AND TRIGGERS

Application/Implementation Notes

There are three basic types of EVENTS within the Synthesis system, those that occur at a specific show time (from the current lock/clock source) in the form MM:SS.FF event parm1 parm2, where the MM:SS.FF field specifies a specific show time (minutes, seconds, and frames) for when the {event} and its associated parameters will execute.

The second type is a special "ON:" event, a system implemented prior to the existence of the far more powerful SUBROUTINES. An "ON:event" is a special, reserved set of event operations that occur during a special condition, such as a system RESET, a "show start", a "show abort", error, end of tape condition, or other defined event occurs. On:Events are limited to 16 "steps" or events per condition, and are preserved for compatibility with previous revisions.

A very powerful portion of the events system is the event "subroutines", which take the form: {Snnn.sss event parm1 parm2}, where the Snnn is a subroutine number from 1 to 250 (allowing up to 250 subroutines), and "sss" is a sequence number that may be used to control the sort-order and clarify the function of the subroutine. The event types and parameters operate exactly the same as timed events. A subroutine may be as many steps as are required for operation; from one up to the maximum number of "events" long.

A subroutine may be directly called by a timed event (using a GOSUB, event type 30) for repetitive or frequent operations. Or it may be linked to occur at a specified number of seconds or frames FROM a previous event or subroutine, using the ArmSec (32) and ArmFrm (33) timing commands, which is very powerful for creating timed sequences, "bucket-brigade" timed events, etc.

The most versatile aspect of subroutines is that they may be associated with an external TRIGGER or event from the outside world (or another controller) that initiates execution of a subroutine as an asynchronous, externally controlled event. To do this, a subroutine may be associated with an external trigger, which would normally be presented as a contact closure, digital input signal, or a serial message from some other device.

On Triad controllers, digital inputs are accomplished through a TC-326 16 Channel Input module. Each channel is a discrete, opto-isolated, independent input bit, and up to four modules are allowed for a total of 64 digital inputs or "triggers". Further, as we monitor TRANSITIONS, it is possible to trigger on the falling (active) OR rising (inactive) edge, or BOTH for each input channel. The ArmEvt (type 31) event associates a TRIGGER with a SUBROUTINE. The first parameter is the SUBROUTINE number, from 1 to 250. Obviously, the subroutine must exist if it is to be actually executed. The second parameter is the trigger number, as defined below. Entering a trigger number of 0 will effectively DISARM the subroutine from any trigger (although the subroutine still exists, and may be later reassociated with another trigger or called by a direct or timed routine). Likewise, using a subroutine number of "0" for a trigger will disarm the specified trigger to (any) subroutine until re-armed.

The TRIGGER numbers are fixed by the hardware, and are defined as follows

01-16	Active triggers from channels 1-16 of the first input card
17-32	Active triggers from channels 1-16 of the second input card
33-48	Active triggers from channels 1-16 of the third input card
49-64	Active triggers from channels 1-16 of the fourth input card
65-80	Inactive edge triggers from channels 1-16 of the first input card
81-96	Inactive edge triggers from channels 1-16 of the second input card
97-112	Inactive edge triggers from channels 1-16 of the third input card
113-128	Inactive edge triggers from channels 1-16 of the fourth input card

To simulate a trigger, or pass a message from an external controller through the RS-232 serial port, an ASCII string consisting of the numeric digits of the trigger number, followed by a "T" (uppercase T) may be used. All numeric digits are parsed up to the "T", which then simulates or executes the trigger based

on the cumulative value of ASCII digits received. Further, trigger values from 129-255 may be used for armed events that are NOT triggered by hardware inputs, rather than exclusively through serial messages. Currently the ASCII messages are only interpreted from the TCOM1: (terminal) serial port. For example, to send a trigger MESSAGE to the LDC/SCU, the following string would send a trigger of 134:

134T

(That's it, NO carriage return, line feed, or other control codes.) This sequence may be entered from a terminal/(emulator) while the system is running. A nnnT trigger message will be displayed if the real-time status display is enabled.

"X" VARIABLE CONTROL

Beginning with release 92.04, it is now possible to pass "X" variable messages to the LDC/SCU system through the serial control port. Two ASCII string numbers must be sent, the first string is the ASCII value, from "0" to "255", followed by a "V" (for value). The second string identifies the "X" variable (legal values are 0 - 63) followed by an ASCII "X" (uppercase). The VALUE MUST be set first, followed by one or more "X" variables to receive the last sent V.alue data. For example, to set X(21)=233, the following ASCII characters would be sent, without control or other codes intervening:

233V21X

(That's it, NO carriage return, line feed, or other control codes.)

UART INITIALIZATION

Several types of UARTS are used based on the processor or I/O board used. The TC-3500 and TC-3510 use Motorola 6850 parts. The TC-3550 Rev. B and C, and TC-3518 Rev. B use Rockwell 65C52 "duart" (dual UART) parts. the TC-3550 Rev. D and E, and BART Controller use the EXAR part. The initialization of the UARTS has been moved to an optional table in the EVENTS ROM/RAM area as defined in the memory map in this document.

Following is a listing of the UART initialization bits, and how to configure the baud, bits, parity, etc.

MOTOROLA 6850 STATUS WORD							
7	6	5	4	3	2	1	0
IRQ	PE	OVRN	FRM	CTS	DCD	TDRE	RDRF
RINT	00=RTS- NI	000=7E2	100=8N2			00= /1	
	01 RTS- IE	001=7O2	101=8N1			01= /16	
	10 RTS+ NI	010=7E1	110=8E1			10= /64	
	11=BRK NI	011=7O1	111=8O1			11= RES-	

The baud rate for the 6850 is determined by hardware jumpers.

65C52 ROCKWELL UART							
These values are for the Rockwell 65C52 UART, used on TC-3550 Revision B and C Processor boards, and for TC-3518 Eight Channel Serial boards.							
Control mode							
This is the first entry in the initialization table for each UART.							
7	6	5	4	3	2	1	0
0	0 CDR	0=1 STP	0=NO	BAUD	RATE		
	1 ACR	1=2 BIT	1=ECHO	0000 =	50	1000 =	2400
				0001 =	110	1001 =	3600
				0010 =	134	1010 =	4800
				0011 =	150	1011 =	7200
				0100 =	300	1100 =	9600
				0101 =	600	1101 =	19.2
				0110 =	1200	1110 =	38.4
				0111 =	1800	1111 =	EXT
Both the receive and send ports of the UART follow the selected baud rate.							

Format mode			
This is the second entry for each UART			
1	00=5 01=6 DATA	00=ODD 01=EVN	1=PRTY 1=DTR 1=RTS
	10=7 11=8 BITS	10=MRK 11=SPC	0=NONE 0=low 0=low

- For the TC-3550 board, the communications parameters for the processor and all of the ports on a TC-3518 serial board may be set under the "C.onfig C.omm" menu.

Please refer to the Reference Section of the manual for the chart showing the complete Rockwell UART port configuration.

Baud Rates are in the low nibble of 3FC0, 3FC2, 3FD0, 3FD4, 3FD8, 3FDC, 3FE0, 3FE4, 3FE8 and 3FEC.

The Rates represented by these values are:

Control Mode Byte Values

00 = 50
01 = 110
02 = 134
03 = 150
04 = 300
05 = 600
06 = 1200
07 = 1800
08 = 2400
09 = 3600
0A = 4800
0B = 7200
0C = 9600 *
0D = 19.2
0E = 38.4
0F = Extn

High Nibble of the Same Byte

00 = No Echo, 1 Stop Bit, CDR *
10 = Echo, 1 Stop Bit, CDR
20 = No Echo, 2 Stop Bits, CDR
30 = Echo, 2 Stop Bits, CDR
40 = No Echo, 1 Stop Bit, ACR
50 = Echo, 1 Stop Bit, ACR
60 = No Echo, 2 Stop Bits, ACR
70 = Echo, 2 Stop Bits, ACR
(The high bit should remain unused)

So, if 3FC0 has a value of 0C, the terminal port is set to 9600, No Echo, 1 Stop Bit and CDR. If the value is 2D, the terminal port is set to 19.2k, No Echo, 2 Stop Bits and CDR.

* Denotes Default Value

For TC-3550E and TC-550 (BART) boards, the parameters begin at 3FC0 and use the following:

EXAR 88C681/68C684	
Offset 1 in the table is the baud rate for the receiver (high nibble) and transmitter (low nibble). Normally these are both set to the same value. I.E. 9600 = $BB	
	Set 1
0000 =	50
0001 =	110
0010 =	134.5
0011 =	200
0100 =	300
0101 =	600
0110 =	1200
0111 =	1050
1000 =	2400
1001 =	4800
1010 =	7200
1011 =	9600
1100 =	19.2/38.4
1101 =	Timer
1110 =	CLK/16
1111 =	CLK/1

Offset 2 is the Mode Register 1, with critical bits:				
4	3	2	1	0
00	w/parity	0 even	00	5 bits
01	force	1 odd	01	6 bits
01	no parity		10	7 bits
			11	8 bits

The default we use is $13, which is 8 bits, no parity.
Leave the high 3 bits OFF!

Offset 3 is the Mode Register 2 controlling the number stop bits. The common values are for 1 or 2 bits:
3 2 1 0
0111 = 1 stop bit
1111 = 2 stop bits
The default we use is $07, or 1 stop bit

As an example, to set the aux comm port on a TC-3550E, the following would be representative:

3FC4: BB 13 07 00

3FC4 is the location of the COM port inits (see memory map)
BB is 9600 baud for both the transmitter and receiver
13 sets up for 8 bit, no parity
07 sets up for 1 stop bit
00 is a reserved variable in the table

EVENT FORMAT/MEMORY MAPPING

Events are normally created and edited from within the Synthesis Show Programming software, which runs on an IBM/Compatible computer. The events data is then transferred to the LDC/SCU/BART Controller either using serial communications when transferring to RAM, or by burning an EPROM and installing the EPROM onto the processor board.

Event data consists of four major types and areas, as follows:

MM:SS:FF EVENTS	Standard, sequential events (six bytes/each).
SUBROUTINES	Those numbers (61:XX:XX to 99:XX:XX) which may be called by another event, trigger, or delay. The seconds and frame fields are used for sorting.
ON:EVENTS	Special event sequences for reset, start, abort, etc.
MACRO STRINGS	ASCII character sequences used for special device control, e.g. laser disc players, video wall and slide controllers, laser controllers, etc., as edited under E.vents-M.acros.

Select E.vts to edit the starting locations, number of events, and target locations.

The current mapping for these areas in a TC-3500/3550 CPU is as follows:

$2000-$2FFF	Standard events	
$3002-$35FF	The subroutine search starts at $2000, and is set based on the division between timed and subroutine event types.	
$3600-$37FF	ON:events consisting of 16 3-byte sequences each	
$3600	ON:reset	Performed on a system reset
$3630	ON:start	Performed by an external start trigger
$3660	ON:abort	Performed by an external abort/show stop trigger
$3690	ON:EOT	Performed if EOT flag set and loss of show data/code
$36C0	ON:END	Performed if END flag set and loss of show data/code
$3800-$3EFF	56: 32 byte macro ASCII sequences	
$3F00-$3F3F	Configuration for analog modes (8/12 bit, etc.)	
$3FF0-$3FFF	Configuration table for custom or special installations.	

UPLOADING EVENTS FROM SYNTHESIS

LDCLOAD Rev 1.0 04/30/91 TWR

During the show programming process it will be necessary to program an events file for the Laser Disc Controller (LDC) and then download these events to the LDC. The following procedure explains how to download events from the CPU (main computer) to the LDC.

First of all keep in mind that most shows have two events files. The main events file that resides in the CPU during the show and the LDC events file which is edited on the CPU and then downloaded to the LDC. Whenever you work with the LDC events file you must remember to re-load the CPU events file before running the show.

The first step in this process is to figure out the filenames for our two events files. For a particular show these may be obtained by looking at the D.isk subsystem screen or by requesting the current names from a Triad staff person. Normally the name of the CPU events file will be "EVENTS.EVT" or something close to this. The name of the LDC events file will be named "EVENTS.LDC", "SHOWLDC.EVT" or something similar to this. If you have any questions regarding the file names to use please consult the Triad staff.

Once you have your filenames in hand follow the following procedure to download the events file to the LDC.

Type the following from the main menu:

KEYSTROKE	EXPLANATION
s	for sync
e	for events
shift l	to load a new events file
y	for yes
showldc.evt	for the file name (this is the LDC events file name)
<enter>	

The events file for the LDC is now loaded in the CPU.

Escape back to the main menu and get ready to upload to the LDC. Type:

KEYSTROKE	EXPLANATION
o	for operator
u	for upload
t	for terminal
<esc>	Hit the escape key once. You should see a menu on the screen that begins with “SCU:”. If you don’t see the SCU menu please consult the Triad staff to reconfigure the terminal mode of the CPU.
e	for events
c	for clear
Y	(Must be capital) for yes. (This will clear any old events.)
<esc>	to get back to the main SCU menu

Now hit <esc> several times (relatively quickly) until you see the display:

```
TRIAD 65
>
```

<alt> x	to get out of the terminal program
g	for go to download events
<enter>	to start the download (this will take about 30 seconds)
<esc>	to get back to main menu.

Now we must re-load in the events file for the CPU. From the main menu type:

KEYSTROKE	EXPLANATION
s	for sync
e	for events
shift l	for load
y	for yes
events.evt	for the file name (this is the filename for the CPU events file)
<enter>	to load the file
<esc>	to get back to the main menu

This should complete the process for downloading the events file to the LDC.

If you have problems feel free to contact Triad for assistance.

Triad Productions
1910 Ingersoll Ave.
Des Moines, IA 50309
voice: 515-243-2125
fax: 515-243-2055
BBS: 515-243-2176 (leave e-mail for the SYSOP)

EVDUFER

Special information for event uploads using TC-3550 Processor/Firmware

Please note:

In the TC-3550 version of the LDC/SCU software, a new method was implemented to define the starting location of event subroutines. In all previous versions, Event subroutines were "hard-coded" to begin at $3000. This limited the number of events used in subroutines to a maximum of 256. In the TC-3550 version, an automatic scan is performed to find the base of subroutines, based on 6 bytes per event. As $3000 does not fall on an even-six boundary (starting with location $2000), it is necessary to load the subroutines starting at a location that is a multiple of 6, i.e. $3002. This number value may be set in the UPLOAD menu, and will hopefully be calculated automatically working down from the top event sub. If more than 256 event subs are required, this number can work down as needed.

In "randomly" initialized RAM memory, all events should be loaded (at least once) to ensure that a bogus subroutine is not accidentally found as the subroutine base. (In an EPROM this is not a problem, as all unused bytes are $FF). If RAM is being used, do a "C" (clear all events) from the EVENT: menu.

The "default" upload address is $2600 for subroutines, which leaves room for ___ timed events and ___ subroutines. This number can be adjusted for the nature of a specific application; an events script that depends heavily on subroutines may load at location $2060, conversely a script that is highly time-dependent may load subroutines much higher. Please contact Triad for further advice or information on selecting addressing parameters.

NOTES

After a JAM CLOCK (event type 38), the FIRST physical event (000) will not be executed. Therefore it is suggested at 00:00:00 a 0 0 0 event be left as the first physical event. This is to allow a one-time initialization event or subroutine call to be performed upon a master reset, but not between show cycles or time code sync operation.

All menus in the SCU/LDC software currently only recognize upper case alpha characters. Unless performing macro edits, it is suggested that the CAPS LOCK function be used on the terminal or emulator being used.

- This has been cleaned up. Most requests and menu options now will automatically convert lower case entries to upper case (except when entering ASCII data (event macro string) or in response to "Are you Sure?").

Please note the difference between EVENT SUBROUTINES, located in the events data area, ASCII STRING MACROS (which are character strings sent to the display or comm ports), and ANIMATION CUE MACROS (also sometimes referred to as animation SUBROUTINES) which are located and processed in an entirely different context.

Animation macros or subroutines are programmed in REAL time, but with the ability to be invoked one or more times at any relative time within a show cycle. Animation macros (/subroutines) are always started (type 36) and may optionally be killed (37) by the EVENTS system. EVENT SUBROUTINES may in turn be invoked at any time from animation macros in addition to direct calls (type 30), armed/triggered calls from the real world (type 31), or after a time delay (type 31).

Although very powerful in a potential application, an in-depth understanding of the difference and use of each type and subsystem is required.

Note that this system is subject to ongoing enhancements and changes due to new features, bug-fixes, and/or changes in the Synthesis programming system. Please verify the revision code and use the commands and documentation that apply to that release, or update the firmware on the processor board.

We believe this information to be correct, and assume no liability for damages or injury (consequential or inconsequential) incurred by the use or application of this software. Please report any errors or omissions in this documentation or the program immediately to Triad Productions, Inc.

SYNTHESIS LDC/SCU REAL TIME COMMANDS

Rev 1.02 03/30/93 WJS

The following commands are recognized through the main (terminal) serial port as "REAL TIME" operations. Note that the characters are pure 8 bit ASCII, with bit 7 = 0, and no parity. All characters above 128 decimal (hi bit set) are RESERVED for TCOM: format data, and must not be used. DO NOT USE ODD or EVEN PARITY, or allow any "hi-bit" set data to be sent through any of the comm ports! One or two stop bits may be used; in some cases we actually prefer two stop bits to improve synchronization especially at the higher baud rates.

There are many variances/exceptions in the interpretation, based on the IO frame or configuration being used, as noted.

SYNC MODES

I (capital I) - GO/switch to Internal sync, based on the crystal clock, and at the current frame rate (15, 24, 25*, or 30 FPS). The clock will continue running from the last SMPTE or pilot time. Note that for TC-550 (BART) and TC-3550 systems, this will be a C.onfigurable option for 59.94 (TV) or 60 Hz. (real time) sync*.

* Not Implemented at this time

L (capital L) - Switch to PILOT sync, based on the pilot frequency (normally video sync at the BNC connector) at a nominal frame rate of 59.94 HZ., as derived from the composite video or video sync signal.

S (capital S) - Switch to SMPTE sync mode. This can be Triad TC-Format time code through the auxiliary RS-232 serial port (I.E. as translated by a TC-500 or TC-750 Time Code Reader), or from direct SMPTE longitudinal time code for TC-550 BART Controllers.

G (capital G) - Toggle the G.eneration of time code for TC-550 BART controllers only. Whenever a new sync mode is selected, this flag is cleared. Note that due to hardware restrictions, it is not possible to READ/lock to SMPTE AND generate SMPTE at the same time. Thus, SMPTE generation is only possible in internal or pilot modes ONLY. The time code is output on pin 5 of the REMOTE connector, as an open collector isolated output.

"DECK" COMMANDS

These are a carry-over from earlier systems, and apply to TC-500, TC-3500, and TC-550 BART systems only, and are not possible on TC-3550 (any release) card frame systems:

A (capital A) - "Abort"/stop command - Issues a .5 second pulse on the stop/abort output (pin 1 of the remote connector).

P (capital P) - "Play" command - Issues a .5 second pulse on the "PLAY" command output (pin 3 of the remote connector).

R (capital R) - "Rewind" command - Issues a .5 second pulse on the "REWIND" command output (pin 5 of the remote connector).

F (capital F) - "Fast-Forward" command - Issues a .5 second pulse output on the FF command output (pin 7 of the remote connector).

These commands only apply to hardware that supports the "deck" functions. Direct ON/OFF or pulse control of these signals is available as "system" [6] events for all hardware that support these opto-isolated output functions.

CONTROL COMMANDS

Z (capital Z) - Zero/zap all analogs and digitals to the preset/reset state. CAUTION: This is a BUMP change to all channels, and will clear everything!

C Code toggle - This toggles the display between an ASCII display string that is readable in terminal mode and binary "TC" format time code that is used by Synthesis when operating in SMPTE mode. The ASCII format appears roughly as: 00:03.12-PR-------EK- and is updated for every frame. This can be useful (and less annoying) when debugging or running through the terminal emulator. The binary format appears as a continuous stream of seemingly "garbage" characters, but which actually represent the time code information to Synthesis. The default mode is normally set in the Cfig (configuration) or setup menu, using the SWITCH option described elsewhere.

ESC (E.scape) - Three times and you're out; at least back to the SCU: menu. Further "escapes" will be acted upon based on frequency. If "fast" enough (within .5 second), the "Marvin" monitor will be entered for debugging/uploading applications. Otherwise, the SCU/LDC firmware will automatically restart, based upon a home/preset condition, and with time code starting at 00:00.00 (unless updated by incoming SMPTE time code, in SMPTE sync mode).

TRIGGERS/VARIABLES

0-9 (ASCII 0-9) - ASCII characters in this range are "parsed" and built into binary numbers from 0 to 255, for interpretation based on the subsequent ASCII letters.

T (capital T) - "T.rigger" - The number previously built from ASCII numbers are translated into an immediate "TRIGGER" event to be performed, assuming that an associated :ON:EVENT: is in effect; if no related trigger is "armed", the result is a "no-op" or no operation. A received message of "004T" would equate to a trigger (ArmEvt) of "4", as received (but the same as that of an "active" or "on" signal from input trigger 4 from hardware). The trigger number is built as an ASCII/decimal string, in the range of 0 to 63.

Example:

To initiate a subroutine within a BART or LDC controller using an ASCII message string, first create the subroutine itself, i.e.:

```
S150.000  --SB-- 000 000 Testing
S150.001  DigOn  001 001 ReadyLamp (will turn on digital output #1)
```

Next, we need to associate the trigger to the subroutine and enable it:

```
00:01.00  ArmEvt 150 150 Testing
```

The first parameter is the subroutine number, the second is the trigger. It is not necessary for the subroutine to be the same number as the trigger, but it helps to avoid confusion when possible to keep them the same.

Note that the above are all events that are uploaded to and executed by the LDC/SCU or BART controller. To test the routine, enter the "T.erminal" emulator and reset the controller. By typing the string "150T" (upper case "T"), the digital channel (01:01) should turn on.

Now, assume we want to automate this process, so that the show control computer can initiate this trigger at a certain point in the performance. The following will show how this can be accomplished. Note the FOLLOWING events should be in a separate file, and execute on the PC running Synthesis.

There are several ways to create and send the string to the LDC/SCU/BART within Synthesis using the M.acro string editor within the events system:

```
001 150                          T\ the rest is a comment
```

Up to 32 characters are available per string. The "\" terminates what will be sent, and the rest of the line can be used for a descriptive comment.

Assuming the LDC/SCU is connected to the COM1: port of the PC, the following event will send the message from Synthesis to the controller:

```
01:00.00  Com1:  000 001 150T\ the rest is a comment
```

At one minute, the string "150T" is sent to the controller, which in turn will call the subroutine 150, which will turn on digital output 01:01.

This is the method for sending messages FROM the PC to the LDC/SCU/BART. Sending messages the other way is similar, but requires a different message format, as ASCII characters would interfere with the time code strings being sent from the controller.

NOTE: Front panel entry of [up] or [down] on a TC-550 BART controller WILL affect/modify the value of a serial command. Thus, manual trigger entries should NOT be attempted with a system :on-line:, with serial trigger messages used :on line:!

V (capital V) - "V.alue" - The number built from the previous ASCII string is assigned to be a VALUE that will subsequently be associated with an "X" variable. The number or value is in the range of 0 to 255, in decimal.

X (capital X) - The "X" variable defined by the number built from the previous ASCII string will be assigned the VALUE previously built from the ASCII numeric string associated with the V.alue command defined above. Thus, a string of 119V61X would set the X variable 61 to a value (or equate) to 119, all in decimal.

Note that in all cases where a numeric string is "defined" (as a T.rigger, V.alue, or X.variable), the numeric string will continue to build and roll-over (mod 255) until a "target" receptor is defined. In the future, we may extend the "Z" (zero/"Zap") command to FORCE all impending ASCII/build values to 0. This is currently NOT the implemented!

TC-FORMAT DATA

Any data with bit 7 (the high bit set) is proprietary, and reserved as part of the TC-format data stream, and results will be unpredictable or unreliable based on the character data and timing as specified by Triad. Therefore, NO data is allowed or supported with the hi-bit set! (WARNING - DO NOT ALLOW ODD OR EVEN PARITY for any terminal operations!)

For further information on the TC-format protocol, please contact Triad Productions for application/ implementation specifications. This format is proprietary and as such, will not be guaranteed or supported in any way for current or future releases of the operating firmware.

TC-3550 EVENT ROM INSTALLATION PROCEDURE

02/07/95 - BRB

It is possible to place events language script data into the EPROM on the TC-3550 LDC and also the BART Controller.

1. Write and debug LDC Events program.
2. Load the .EVT file into the Synthesis Events Editor.
3. Enter the O.perator U.pload area of Synthesis.
4. Toggle F.mt until it reads Intl.
5. Toggle O.ut until it reads File:
6. Hit G.o. Synthesis will prompt for filename (use .HEX extension).
7. Quit Synthesis. Run EPROM burning software.
8. Burn the EPROM using Intel-HEX format!
9. Clear any events from the LDC's RAM.
10. Turn off power to LDC and remove the TC-3550 from the card frame.
11. Place the EPROM into U11 (the middle socket) on TC-3550.
12. Remove all jumpers (hang them off the ends) from JP4.
13. Replace the TC-3550 in the card frame.
14. Power up and test program.

TC-550 BART EVENT ROM CREATION/INSTALLATION

08/09/95 BRB

1. Write and debug BART Events program.
2. Load the .EVT file into the Synthesis Events Editor.
3. Enter the O.perator U.pload area of Synthesis.
4. Toggle F.mt until it reads Intl.
5. Toggle O.ut until it reads File:
6. Hit G.o. Synthesis will prompt for filename (use .HEX extension)
7. Quit Synthesis. Run EPROM burning software.
8. Load Current BART firmware image.
   Note: Must be special version. (94.9Q only known version)
9. Toggle ROM burning software to HEX format!
10. Load the .HEX file created in steps 1-6 on top of the firmware image.
11. Edit memory locations 71FC0-71FCF so that they are as follows:
    BB 13 07 00 BB 13 07 00 BB 13 07 00 BB 13 07 00
12. Save the resulting file (for future use).
13. Burn the EPROM.
14. Test the EPROM with the ROM burning software.
15. Install the EPROM into the BART.
16. Test the software.

CONVERT BINARY TC EVENTS FILES TO SYNTHESIS EVENTS FILES

It is possible to create a Synthesis events file “EVENTS.EVT” from a BART or LDC Processor in the case that the original .EVT file has been lost. EVTCVT.EXE is a program that converts a binary TC events file into one that is formatted for use by Synthesis. There are several differences between TC events files and Synthesis events files. Basically they are:

DESCRIPTION	TC EVENTS	SYNTHESIS EVENTS
Max # of events	938	1000
Max # of macros	56	128
Bytes per event	6	8
Storage for X var. names	NONE	up to 2560 bytes *
Storage for anim. macro names	NONE	up to 2560 bytes *
Storage for SBR names	NONE	up to 2560 bytes *

* 2560 bytes total are reserved for storage of ASCII names of X variables, animation macros, and subroutines.

First you must capture the events data from the EPROM or Battery Backed up RAM with a terminal program capable of doing an ASCII capture and saving to a Motorola format (.MXT) file such as ProComm or Triad’s RomBake. Capture data between $2000 and $3FFF using the Marvin L)ist command.

It will be assumed that events start at $2000, ON events start at $3600, and macros start at $3800. RomBake will save this data to a Motorola-format file with a default extension of ".MXT". A utility called "S1LOAD" will convert a Motorola-format file to binary using the command line as follows:

```
S1LOAD FILE.MXT FILE.BIN 2000 3FFF
```

Note that both the start and end addresses of the file need to be specified.

The resulting binary file may be used as input for the EVTCVT program as follows:

```
EVTCVT FILE.BIN FILE.EVT
```

FILE.EVT can then be loaded into Synthesis for editing and manipulation.

BURNING CUE ROMS FOR TC-3550’S WITH TC-3505’S

Prepare your cue file as usual, programming, defining macros, etc.

1. From the main menu, select O.perator, then U.pload.
2. Press F for F.ormat until the Frmt: prompt reads "Bin". (four times if it's TC).
3. Press O for O.ut until the Out: prompt reads "File:". (three times if it's Com1:).
4. Press R for R.om and select the appropriate size (32k worked for a 27C256)
5. Press C for C.ues and enter 99999 and press enter twice.
6. Press X to toggle Xchg: to ON.
7. Press D for D.o and then Enter to begin writing the file.

Note: Xchg: being on is very important (at least if the firmware is as old as 93.03).

CONVERT LIGHTING CUES TO ANIMATION CUES

8/28/96 - BRB

At times it is necessary to convert lighting cues to animation cues. This is usually done to allow lighting cues to be developed using the lighting system on the PC and then burned into an EPROM for playback from a stand-alone LDC or BART. It has also been used to allow more lighting cues than could be stored in a single 199 cue lighting file. Since the development of DMX output from BARTs and LDCs, this process will become more useful and familiar. A step-by-step outline of the process follows:

1. Create/Modify Lighting (.lgt) cue file (and have it loaded).
2. Make sure that the event editor has a blank events file loaded.
3. Zap the cue file memory space (using G.oodies/z.ap).
4. Load a channel set -or- create a channel set using the analog channels to be included.
5. Reset from the main menu.
6. Go into the Lighting subsystem and choose O.ther/c.onvert. **NOTE: Convert doesn't appear in the menu and must have SHIFT pressed to access.** This toggles the lighting cue conversion bit. (There will be an orange [CLC] flag flashing in the upper right corner of the display (between the sync mode and the clock).
7. Exit the Lighting subsystem and update events when prompted.
8. Put Synthesis into E.xecute mode.
9. Put the A.nalogs into update.
10. Wait for the clock to run past the end of the last lighting cue and stop.
11. Load Blank Event and Lighting files into the Event and Lighting editors.
12. Test Run to see if the conversion worked.
13. Expand the Cue file or portions surrounding macro ID tags (using G.oodies/C.ue Modify/X.pand).
14. Add Macro IDs (beginning and end).
15. Save Cue File (D.isk/S.ave).
16. Burn EPROM if needed (See SCU Events Section of manual for details).

CHANGES WITH EXPANDED ASCII MACRO SPACE IN LDC/BART

Introduction

The BART and LDC have been limited to the use of 62, 32 byte (see below) or 112, 16 byte macro strings. While this is usually ample for most applications, sometimes this limits the LDC or BARTs ability to do a particular job. While there has been a way to get around this problem, a more permanent solution has been found. This WILL cause some programs written for earlier firmware to break as certain set-up tables are being moved to different memory locations, so read carefully, and pay **CLOSE** attention to firmware versions.

CHANGES TO THE BART/LDC
Rev. 96.10 (and later)

The ATYPE array has been moved from $3F00 to $0280. This is immediately below the preset array located at $02C0.

NOTE: This is very important for POKE operations in systems that use 12 bit or other custom analog configurations !! Be extremely careful to check revisions and addresses when setting up self-poking code!

This move opens up 6 new, 32 byte macro strings for a new total of 62. (Two more will be coming real soon!) The event Macro editor now allows all 64 potential macros to be selected. (Be careful not to step on the comm goodies at $3FC0 - $3FEF.)

Changed event type 18 to "SETPAG". PARM2 sets the base page of 8 consecutive pages of 64: 32 byte macro strings FOR THE NEXT MACRO used. After an ASCII macro is sent, the page returns to the default address ($38, or 54 decimal). This allows for MANY, MANY groups of 64: 32 byte or 128: 16 byte strings, addressed as before in any of the comm or display commands.

(The Setptr -> lo/hi count, string followed by XCOM ppp 000 still will work in memory areas 1800-1FFF or $4000-7FFF.)

For the time being, if you want to use the LDC/BART's built-in editor to edit or play with a base other than $38xx, put a type 18 event in with the page you want to use, but without a subsequent use of any macros. Let it execute, then go into the macro editor! (This will be changed to allow bank selection from the menu.) There is currently no provision made in Synthesis for editing/uploading more than one page of macros per events file. A DOS-based conversion program (which will allow off-line editing/saving of these extended macro pages) is available from Triad. This conversion program will convert a plain ASCII text file into a TC-upload format file set-up to upload to one of four selected address pages (chosen from a menu).

PARM1 is reserved for bank memory control, i.e. P1=0 will select RAM, while other values will switch over to extended RAM or ROM in the extended sockets, all overlapping in the range of 4000-7FFF.

Some good locations to use currently are:

1900 - 1FFF = 56 macros (page 18 for use in BART--PARM2=25)
4000 - 47FF = 64 macros (be careful if using cues, do not use if RAM firmware) (PARM2=64)
4800 - 4FFF = as above (PARM2=72)
7800 - 7FFF = 64 macros, safe with RAM version of firmware (PARM2=120)

The old event type 18 (X(P1) .or. P2) can be handled with the ModifyX events; the only use known was for Intamin/Korea.

CHANGES TO SYNTHESIS (CLASSIC)

G.o now does an automatic "W.akeup" to hopefully ensure that the LDC/BART is in MARVIN. After the BART/LDC goes into terminal mode, the user is given the option of proceeding or aborting the upload. If

the upload is continued, the terminal window is opened, and all characters are echoed back from the BART/LDC (in TCmode) to the screen during an upload. It waits for each character to echo, so the throughput is cut in half but the good news is that it works correctly even with 1 stop bit, so it should prove to be very reliable. You can also follow the progress of the upload.

The target address for event subroutines is now automatically calculated (at the end of time based events + 1). Hopefully this will eliminate a potential "gotcha" caused by trying to load too many subroutines and not changing the $2600 to a lower value. To keep things clean, Triad would recommend an events Z.ap to initialize RAM prior to upload, especially to existing systems.

ON: events have been moved to page $3700!! **CAUTION: BE SURE TO UPDATE THE DEFAULT.DFL TO START ON:EVENTS HERE! ALSO, CHANGE THE NUMBER OF ON:EVENTS TO 80, (NOT 96!)**. You will also probably want to set the default # of ASCII macros up from 56 to 62 to accommodate the new strings.

ROCKWELL UART PORT SETTINGS

The following chart lists all of the possible combinations of port settings along with their decimal and hex equivalents. See SCU Event, UART initialization for further information.

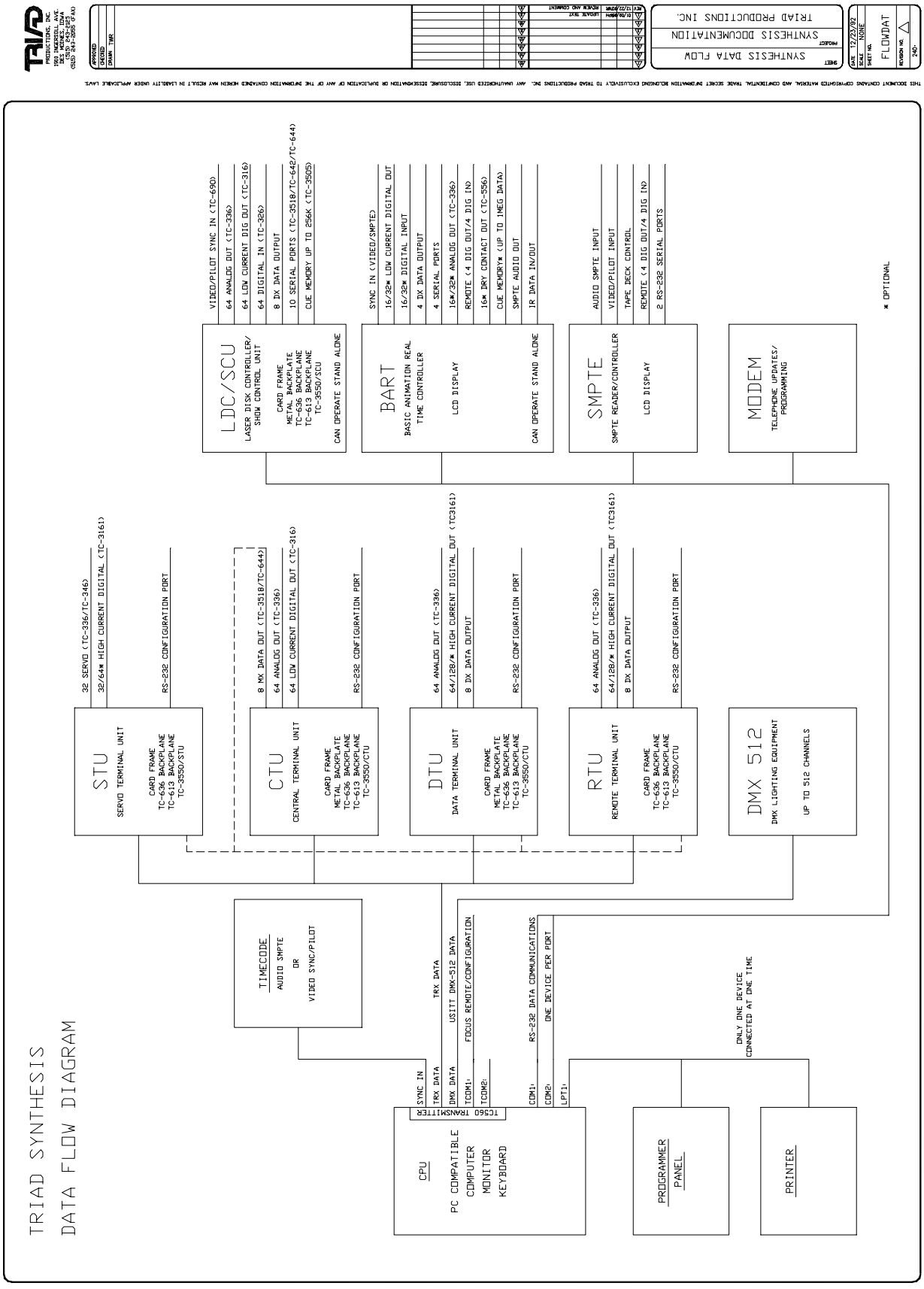
Dec	Hex	Data	P Type	Parity	DTR	RTS
128	80	5	Odd	None	Low	Low
129	81	5	Odd	None	Low	High
130	82	5	Odd	None	High	Low
131	83	5	Odd	None	High	High
132	84	5	Odd	Parity	Low	Low
133	85	5	Odd	Parity	Low	High
134	86	5	Odd	Parity	High	Low
135	87	5	Odd	Parity	High	High
136	88	5	Even	None	Low	Low
137	89	5	Even	None	Low	High
138	8A	5	Even	None	High	Low
139	8B	5	Even	None	High	High
140	8C	5	Even	Parity	Low	Low
141	8D	5	Even	Parity	Low	High
142	8E	5	Even	Parity	High	Low
143	8F	5	Even	Parity	High	High
144	90	5	Mark	None	Low	Low
145	91	5	Mark	None	Low	High
146	92	5	Mark	None	High	Low
147	93	5	Mark	None	High	High
148	94	5	Mark	Parity	Low	Low
149	95	5	Mark	Parity	Low	High
150	96	5	Mark	Parity	High	Low
151	97	5	Mark	Parity	High	High
152	98	5	Space	None	Low	Low
153	99	5	Space	None	Low	High
154	9A	5	Space	None	High	Low
155	9B	5	Space	None	High	High
156	9C	5	Space	Parity	Low	Low
157	9D	5	Space	Parity	Low	High
158	9E	5	Space	Parity	High	Low
159	9F	5	Space	Parity	High	High

Dec	Hex	Data	P Type	Parity	DTR	RTS
160	A0	6	Odd	None	Low	Low
161	A1	6	Odd	None	Low	High
162	A2	6	Odd	None	High	Low
163	A3	6	Odd	None	High	High
164	A4	6	Odd	Parity	Low	Low
165	A5	6	Odd	Parity	Low	High
166	A6	6	Odd	Parity	High	Low
167	A7	6	Odd	Parity	High	High
168	A8	6	Even	None	Low	Low
169	A9	6	Even	None	Low	High
170	AA	6	Even	None	High	Low
171	AB	6	Even	None	High	High
172	AC	6	Even	Parity	Low	Low
173	AD	6	Even	Parity	Low	High
174	AE	6	Even	Parity	High	Low
175	AF	6	Even	Parity	High	High
176	B0	6	Mark	None	Low	Low
177	B1	6	Mark	None	Low	High
178	B2	6	Mark	None	High	Low
179	B3	6	Mark	None	High	High
180	B4	6	Mark	Parity	Low	Low
181	B5	6	Mark	Parity	Low	High
182	B6	6	Mark	Parity	High	Low
183	B7	6	Mark	Parity	High	High
184	B8	6	Space	None	Low	Low
185	B9	6	Space	None	Low	High
186	BA	6	Space	None	High	Low
187	BB	6	Space	None	High	High
188	BC	6	Space	Parity	Low	Low
189	BD	6	Space	Parity	Low	High
190	BE	6	Space	Parity	High	Low
191	BF	6	Space	Parity	High	High

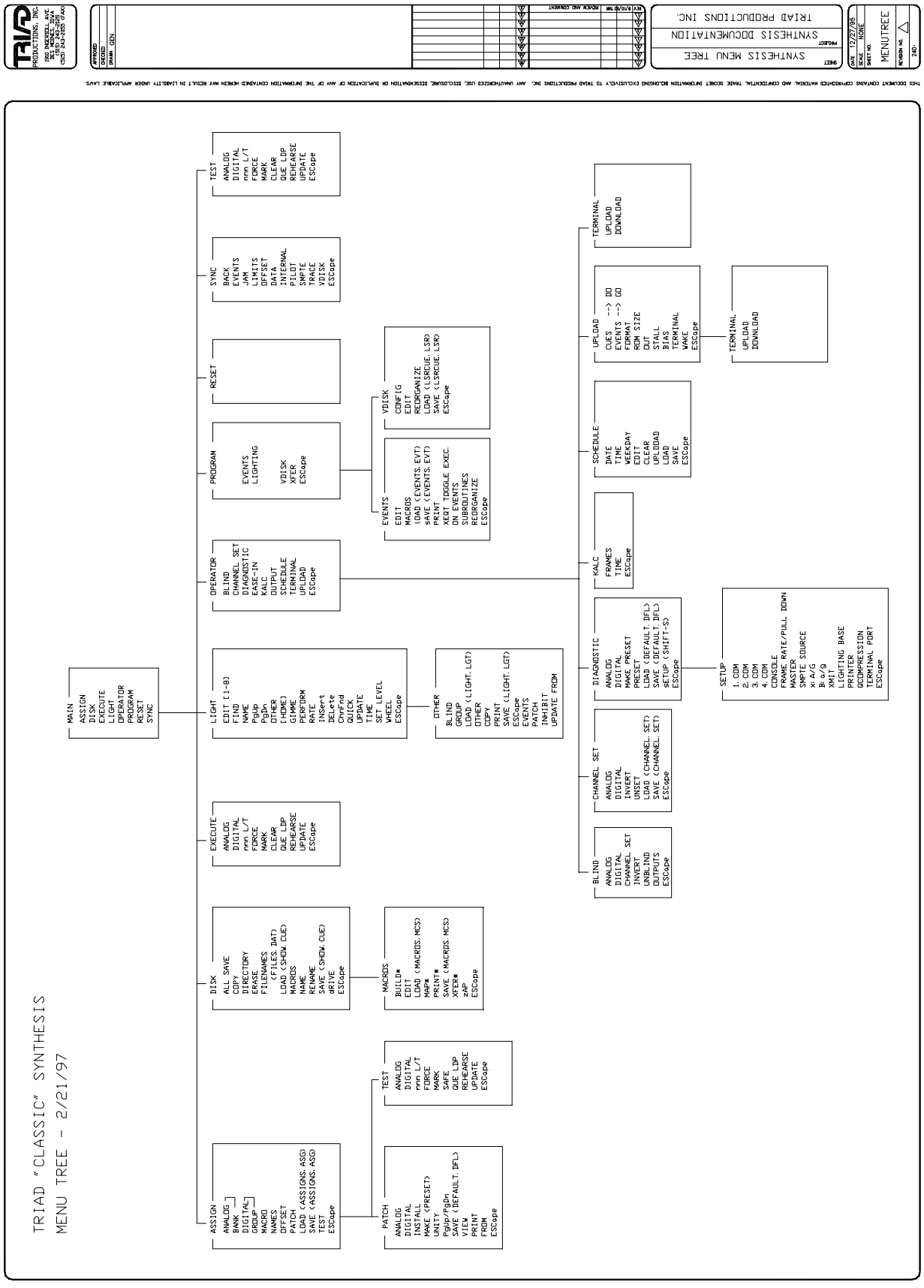
Dec	Hex	Data	P Type	Parity	DTR	RTS
192	C0	7	Odd	None	Low	Low
193	C1	7	Odd	None	Low	High
194	C2	7	Odd	None	High	Low
195	C3	7	Odd	None	High	High
196	C4	7	Odd	Parity	Low	Low
197	C5	7	Odd	Parity	Low	High
198	C6	7	Odd	Parity	High	Low
199	C7	7	Odd	Parity	High	High
200	C8	7	Even	None	Low	Low
201	C9	7	Even	None	Low	High
202	CA	7	Even	None	High	Low
203	CB	7	Even	None	High	High
204	CC	7	Even	Parity	Low	Low
205	CD	7	Even	Parity	Low	High
206	CE	7	Even	Parity	High	Low
207	CF	7	Even	Parity	High	High
208	D0	7	Mark	None	Low	Low
209	D1	7	Mark	None	Low	High
210	D2	7	Mark	None	High	Low
211	D3	7	Mark	None	High	High
212	D4	7	Mark	Parity	Low	Low
213	D5	7	Mark	Parity	Low	High
214	D6	7	Mark	Parity	High	Low
215	D7	7	Mark	Parity	High	High
216	D8	7	Space	None	Low	Low
217	D9	7	Space	None	Low	High
218	DA	7	Space	None	High	Low
219	DB	7	Space	None	High	High
220	DC	7	Space	Parity	Low	Low
221	DD	7	Space	Parity	Low	High
222	DE	7	Space	Parity	High	Low
223	DF	7	Space	Parity	High	High

Dec	Hex	Data	P Type	Parity	DTR	RTS
224	E0	8	Odd	None	Low	Low
225	E1	8	Odd	None	Low	High
226	E2	8	Odd	None	High	Low
227	E3	8	Odd	None	High	High
228	E4	8	Odd	Parity	Low	Low
229	E5	8	Odd	Parity	Low	High
230	E6	8	Odd	Parity	High	Low
231	E7	8	Odd	Parity	High	High
232	E8	8	Even	None	Low	Low
233	E9	8	Even	None	Low	High
234	EA	8	Even	None	High	Low
235	EB	8	Even	None	High	High
236	EC	8	Even	Parity	Low	Low
237	ED	8	Even	Parity	Low	High
238	EE	8	Even	Parity	High	Low
239	EF	8	Even	Parity	High	High
240	F0	8	Mark	None	Low	Low
241	F1	8	Mark	None	Low	High
242	F2	8	Mark	None	High	Low
243	F3	8	Mark	None	High	High
244	F4	8	Mark	Parity	Low	Low
245	F5	8	Mark	Parity	Low	High
246	F6	8	Mark	Parity	High	Low
247	F7	8	Mark	Parity	High	High
248	F8	8	Space	None	Low	Low
249	F9	8	Space	None	Low	High
250	FA	8	Space	None	High	Low
251	FB	8	Space	None	High	High
252	FC	8	Space	Parity	Low	Low
253	FD	8	Space	Parity	Low	High
254	FE	8	Space	Parity	High	Low
255	FF	8	Space	Parity	High	High

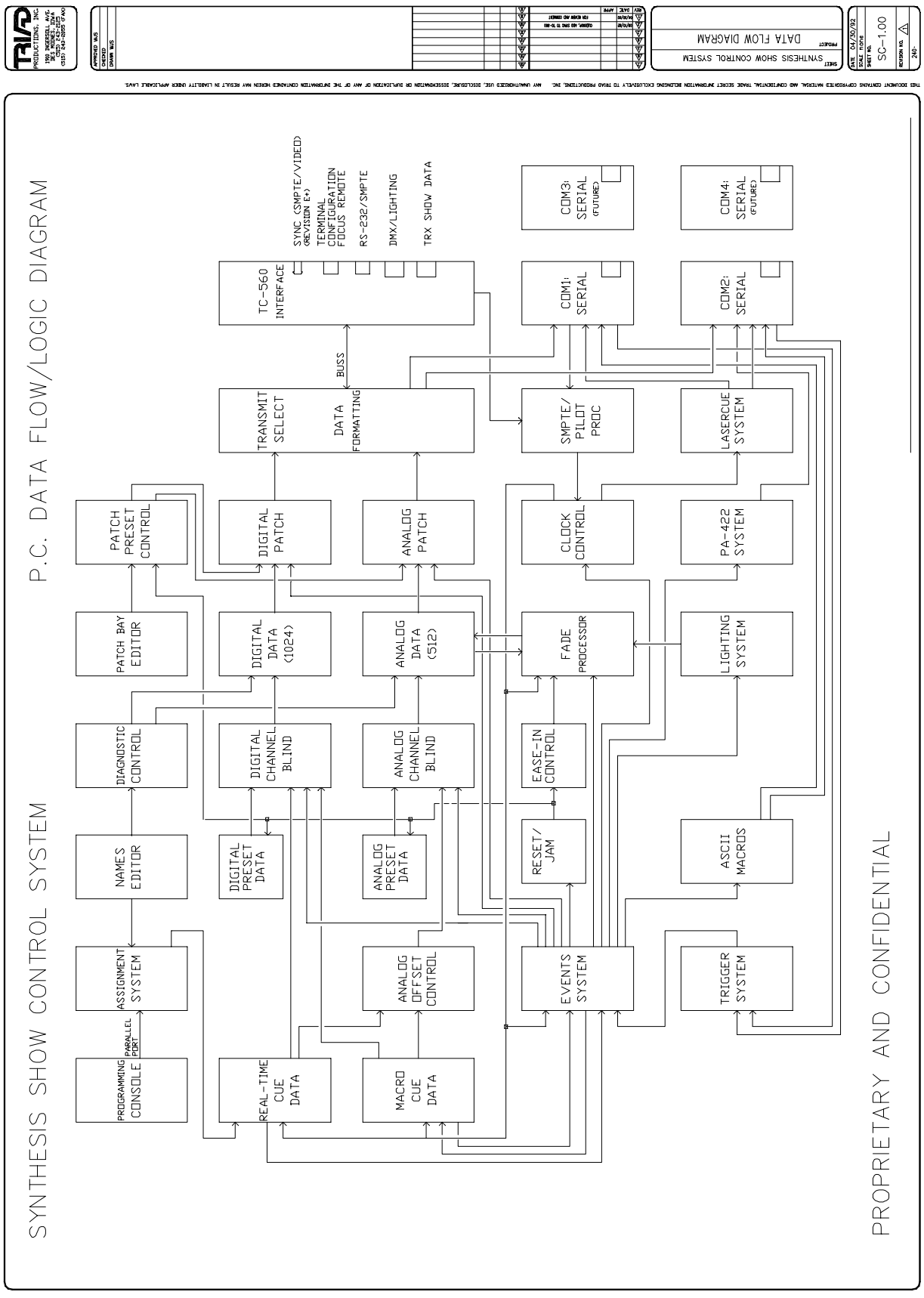
FLOWDAT SYNTHESIS DATA FLOW



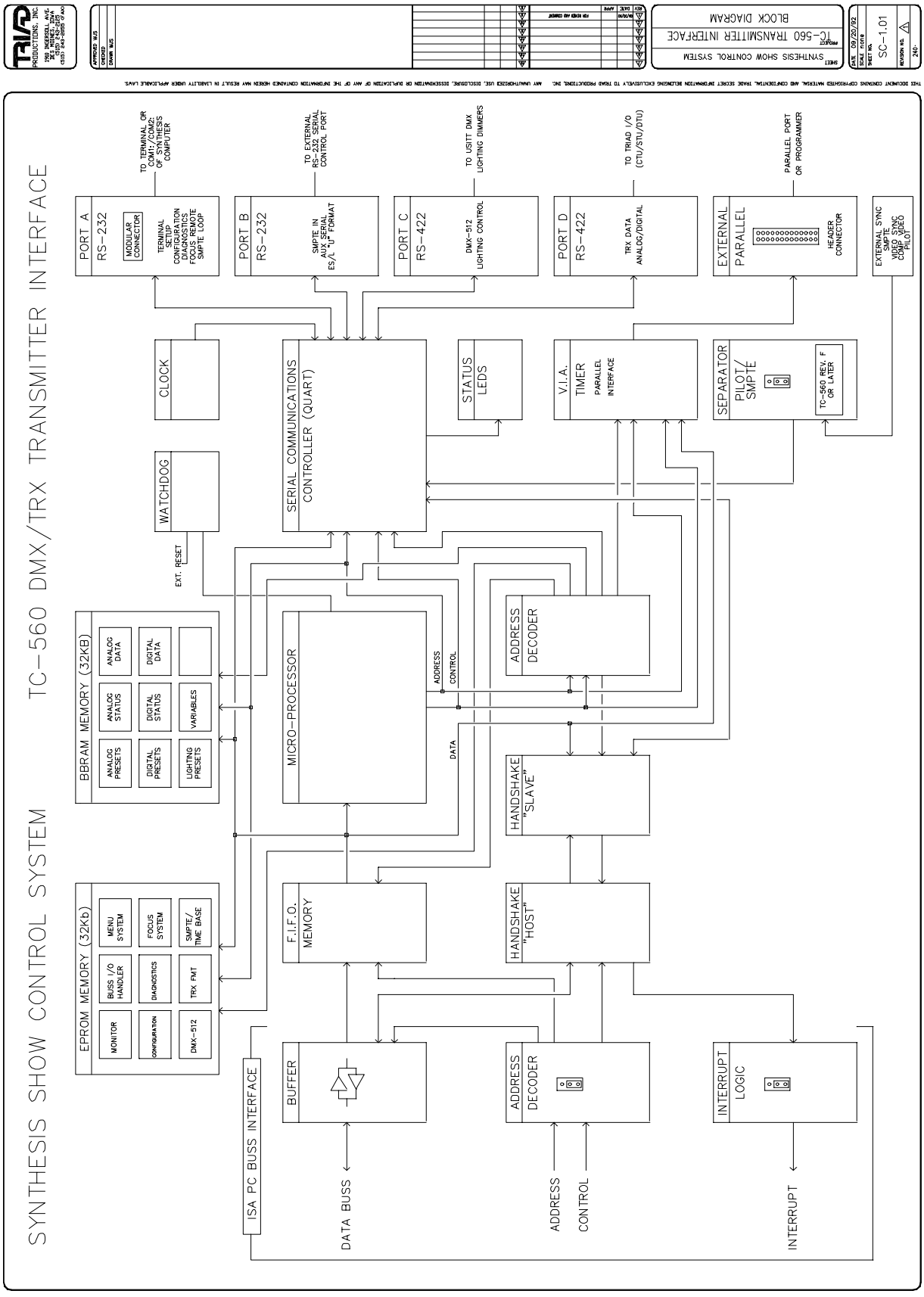
MENUTREE SYNTHESIS MENU TREE



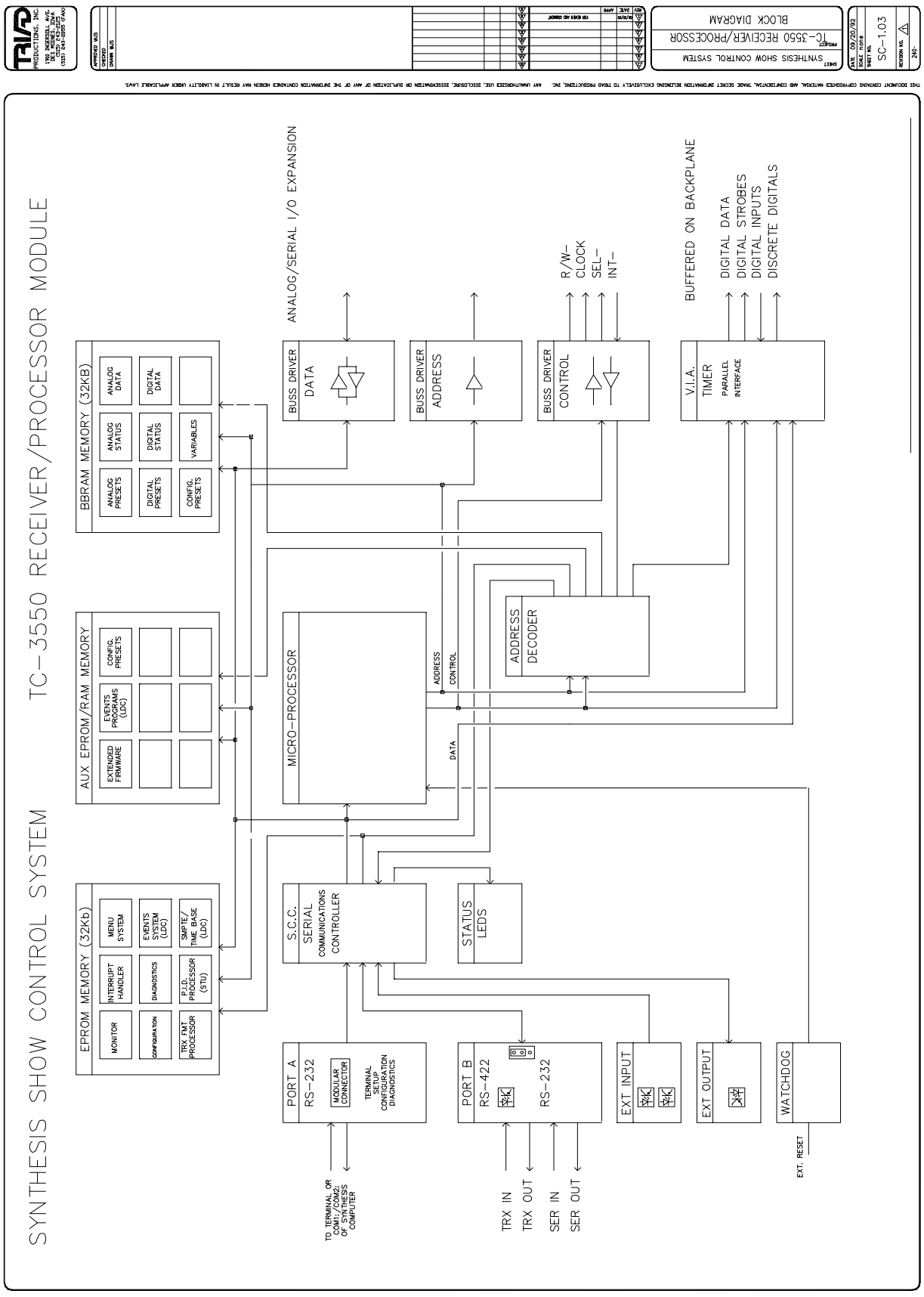
SC-1.00 PC DATA FLOW/LOGIC DIAGRAM



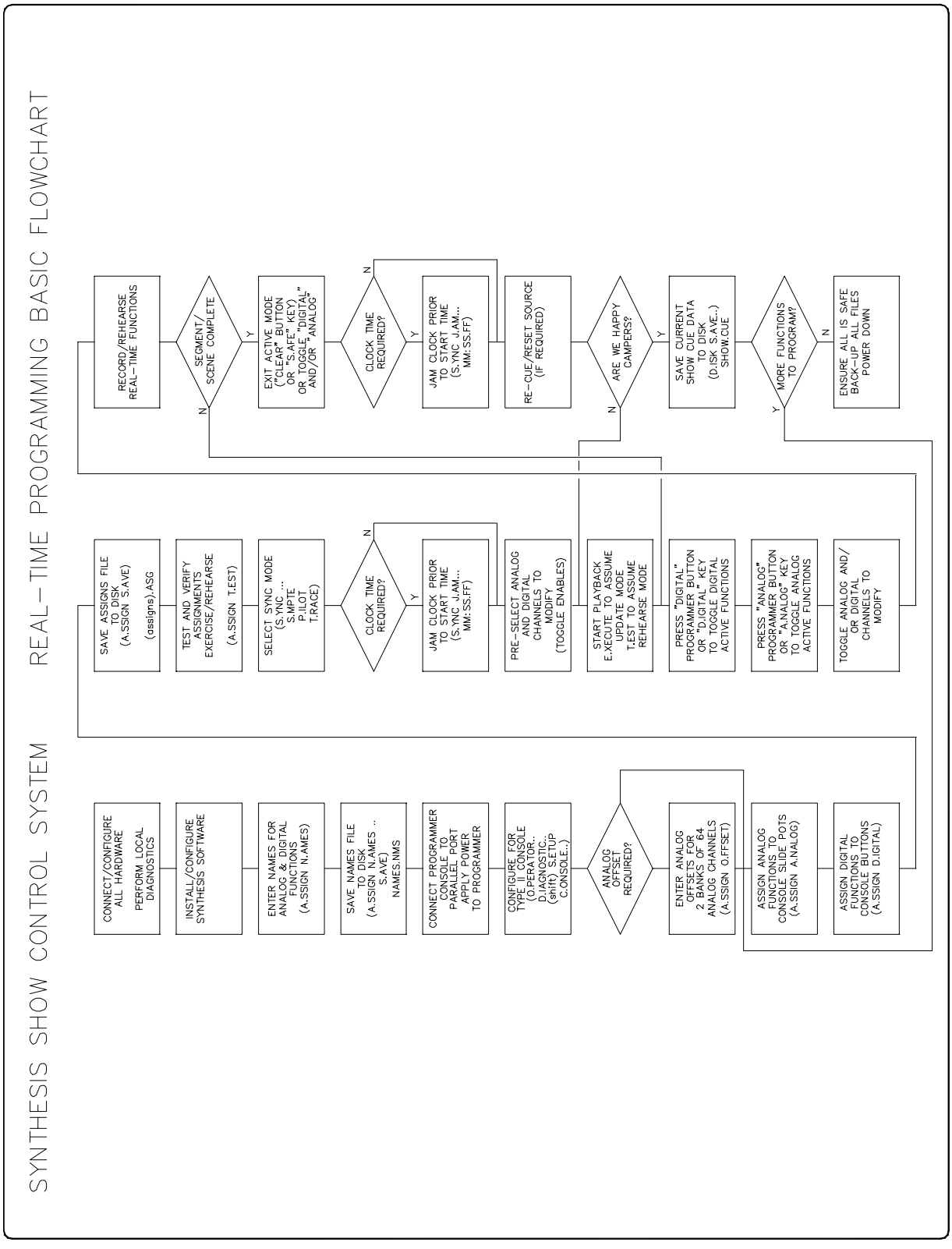
SC-1.01 TC-560 DMX/TRX TRANSMITTER INTERFACE



SC-1.03 TC-3550 RECEIVER/PROCESSOR MODULE



SC-1.11 REAL-TIME PROGRAMMING BASIC FLOWCHART



SC-1.13 REAL-TIME CUE MACRO CREATION FLOWCHART