9500 Keyboard Commands
1/15/2012 - Michael Seedman

These commands are used from a terminal connected to the 9500
Terminal and Amplifier Baud Rate is 115,200 bps, no parity, 8 bits, 1 stop bit (115200,N,8,1)
You can hit the "ENTER" key a few times to make sure you're speaking to the amplifier

These are NOT used in normal operation, but if you're writing code to interface with the 9500, these are the commands and responses
There are TWO formats for commands:

single key commands that set or toggle parameters in the amp
multiple key commands that set parameters to a certain value
Note: Some commands need to be in "password" mode - this is documented in the command list

There are TWO separate parsers and you get to the second by issuing a '#' in the first

These are the main parser commands and results. These are used in the factory to set up an amplifier
The Alternate parset commands are listed below this list

MAIN PARSER COMMANDS AND RESULTS

Command	Result
~	Reset Master Controller
#	Change parser to Parser2
0	Generate an $APA00 string
1	Outputs: Bandset, FaultHold, BandChanging, PullPhase, Keysense
2	Generate an $APA02 string
3	Generate an $APA03 string
4	Generate an $APA04 string
5	Generate an $APA05 string
6	Generate an $APA06 string
7	Generate an $APA07 string
8	Generate an $APA12 string
9	Generate an $APA11 string
!	Generate an $APA13 string
a	Outputs: ; AutoState, POF[0], OutPfThr
b	Toggles Key on and off
d	Outputs: Dump Eeprom
e	Toggles External Fan
f	Outputs: Frequency this band
g	Outputs: Autotune History
o	Decrements I2C Baud Rate Set and Outputs: I2C Baud Rate
p	Increments I2C Baud Rate Set and Outputs: I2C Baud Rate
r	Cycles through latching relays and Outputs: Relay:
v	Set Wattmeter Defaults
w	Outputs: InVfRaw, InVrRaw, OutVfRaw, OutVrRaw, PfD
x	Disable Serial1 command processor
B	Force Band Change
C	Save Wattmeter cal coeffecients if PW is enabled
D	Set Default calibration values
E	Copy Defaults to user memory User1 and User2
F	Outputs: NewFrequency from counter, New Band, OR Invalid Frequency
K	Outputs: Cath. Board IN or OUT
L	Set Load Cap from Serial Input data 0-100
O	Toggle Echo Button
P	Password Entry
R	Clear Fault Log and Output: Logs
S	Set Serial Number
T	Set Tun Cap from Serial Input data 0-100

X	Save Band Pot setting
Y	Manual/Auto tap - this calls a routine to set the tap manually if necessary
Z	Write Calibration to EEPROM
>	Increment chosen parameter
<	Decrement chosen parameter
.	Choose next parameter in list
,	Choose previous parameter in list
+	Toggle I2C_Debug
/	Outputs: bandchanging, keysense, bandset
_	Toggle SBFlag
*	Clear abuse Log
;	Sets Cal increment = 1, and Outputs: Cal increment
'	Sets Cal increment = 100, and Outputs: Cal increment
^	Toggles ignoreBand
]	Move to next state, and Outputs: state
[	Move to nex lower state, and Outputs: state

ALTERNATE PARSER COMMANDS AND RESULTS

These are Parser2 commands and results. These are used in the factory to set up an amplifier

These commands are used from a terminal connected to the 9500
Terminal and Amplifier Baud Rate is 115,200 bps, no parity, 8 bits, 1 stop bit (115200,N,8,1)
You can hit the "ENTER" key a few times to make sure you're speaking to the amplifier

These are NOT used in normal operation, but if you're writing code to interface with the 9500, these are the commands and responses

Routines to process these commands start with a $
Format is $tt, param1, param2,…paramN<cr>
You enter this mode from the terminal by typing a '#'

Command	**Result**
$00	Telemetry Request, Parameter is type of telemetry requested
$01	Simulate Button Press, Parameter is the button number
$02	Password Challenge
$03	Store byte in eeprom, Parameter is address, data. Only write if amp in state 0
$04	Clear fault log, no parameter
$05	Reset Master Controller, no parameter
$06	EEPROM dump, no parameter. Only works if amp is in state 0
$07	Turn off Legacy Serial Processor
$08	Calibration request, tlmAction, AdjParameterNumber, Value
$09	Clear Abuse Log
$0A	Not Implemented
$0B	Not Implemented
$0C	Not Implemented
$0D	Not Implemented
$0E	Not Implemented
$0F	Not Implemented

9500 Return Strings and Format

The serial port continuously outputs data at 115,200 bps, using 8 data bits, 1 stop bit, and no parity (N,8,1)
All characters are “human readable”, i.e. they will be displayed as ASCII.

A complete set of data output for one “measurement” consists of a data “sentence”. Each sentence is identified with a start, or sentinel, character "$"
Each sentence is subsequently terminated with an end character (*) plus two additional characters in Hex. This is the Checksum of that line. Each sentence is separated by a <carriage return> and <line feed>
additional characters (FF). Each sentence is separated by a <carriage return> and <line
Each sentence consists of a variable number of words and each word in a “sentence” is separated by a comma
The identifier letters in the amplifier are always "APA" and one two digit number defining the kind of telemetry string being sent
A typical First Word in a sentence could appear as follows (including the comma separator):
$APA00,Serial Number, ESN, Version, Mains Board Version, Display Controller Version, Stepper Motor Version, Sound Generator Version, *, checksum
There are currently 12 "$APA" strings that the Amplifier generates

Routine Description	**Routine Name**
Make Serial Number Sentence	MakeTlm0
Make Basic Operational Telemetry Sentence	MakeTlm2
Make Misc Telemetry Sentence	MakeTlm3
Makeband limits Telemetry Sentence	MakeTlm4
Make Button Telemetry Sentence	MakeTlm5
Make Fault Telemetry Sentence	MakeTlm6
Make Segment frequencies Telemetry Sentence	MakeTlm7
Make Band/Seg/Tune/Load Telemetry Sentence	MakeTlm8
Send Raw AD Values Telemetry Sentence	MakeTlm9
Send Block EEPROM Telemetry Sentence	MakeTlm10
Send Common Parameters Telemetry Sentence	MakeTlm11
Send Current Wattmeter Scaling Coef	MakeTlm12
Send Raw Wattmeter AD Counts Telemetry Sentence	MakeTlm13

Routine Description	**Concatenated String Examples**
Make Serial Number Sentence	$APA00,SerialNumber[0:11],ESN(Hex)[0:4]+CH1+CH2,Version(Hex)[0:4]+CH1+CH2,MainsBoardSWVer(Hex)[0:3]+CH1+CH2,DisplayControlerSWVer[0:3],StepperMotorSWVer[0:3],SountGenSoftwareVer(Hex)[0:3]+CH1+CH2,*,Checksum
Make Basic Operational Telemetry Sentence	$APA02,ForwardPower[4:0],SWR (tents)[2:0],InputPower[3:0],HighVoltage(volts)[3:0],PlatCurrent(mA)[3:0],Gain(tenths)[2:0],GridVoltage(-1*tenths)[2:0],GridCurrent(ma)[2:0],Band,AmpState,FaultCode[1:0],KeySense,ForwardPowerPEP[4:0],Checksum
Make Misc Telemetry Sentence	$APA03,Plus5(tenths)[2:0],Plus12(tenths)[2:0],Plus24(tenths)[2:0],Minus12(tenths)[2:0],Plus40(tenths)[2:0],ACLine(tenths)[3:0],MainsBoardStatus[2:0],MainsBoardTap,TempSign[1]+Temp[2:0]+"."+TempDecimal,ExternalFanState,*,Checksum

Makeband limits Telemetry Sentence	$APA04,loop 17 times generating edgeTable[4:0],*,checksum
Make Button Telemetry Sentence	$APA05,Band[1]+Segment[1],Antenna (Hex) +"H",LED_Dim+LED_Snd+LED_PEP+LED_Del+"H"+State,WarmupTime(Hex) +CH1+CH2,TuneValue(Hex)+CH1+CH2,LoadValue(Hex) +CH1+CH2,DisplayedFault(HEX)+CH1+CH2,AutoTuneState,WasteHeat(watts) [3:0],IgnoreBand,ShowHV/WasteHeat,*,checksum
Make Fault Telemetry Sentence	$APA06,Log Entries loop Byte(Hex)+CH1+CH2,Log Entries loop Flog(Hex) +CH1+CH2,*,checksum
Make Segment frequencies Telemetry Sentence	$APA07,Band,Segment 0 Freq[4:0],Segment 1 Freq[4:0],Segment 2 Freq[4:0],Segment 3 Freq[4:0],Segment 4 Freq[4:0],*,checksum
Make Band/Seg/Tune/Load Telemetry Sentence	$APA08,Band,Segment,TuneCapSetting[2:0],LoadCapSetting[2:0],FreqByte1(Hex) +CH1+CH2,FreqByte0(Hex)+CH1+CH2,FreqCounterValid,*,checksum
Send Raw AD Values Telemetry Sentence	$APA09,InVfRaw[3:0],InVrRaw[3:0],OutVfRaw[3:0],OutVrRaw[3:0],VgRaw[3:0],IgRaw[3:],*,checksum
Send Block EEPROM Telemetry Sentence	$APA10,TempWord1(Hex)+CH1+CH2,TempWord0(Hex)+CH1+CH2,16 Eeprom Values(Hex),CH1+CH2,*,checksum
Send Common Parameters Telemetry Sentence	$APA11,IgM[2:0],IgB[2:0],VgM[2:0],Grid[2:0],I2C Baud Rate Divisor[2:0],*,checksum
Send Current Wattmeter Scaling Coef	$APA12,Band,InputWattmeterForwardPowerSlope[2:0],InputWattmeterInterceptSignBit,InputWattmeterForwardPowerIntercept[2:0],InputWattmeterReversePowerSlope[2:0],InputWattmeterInterceptSignBit,InputWattmeterReversePowerIntercept[2:0],OutputWattmeterForwardPowerSlope[2:0],OutputWattmeterInterceptSignBit,OutputWattmeterForwardPowerIntercept[2:0],OutputWattmeterReversePowerSlope[2:0],OutputWattmeterInterceptSignBit,OutputWattmeterReversePowerIntercept[2:0],*,checksum
Send Raw Wattmeter AD Counts Telemetry Sentence	$APA13,Band,InputWattmeterForwardCountsRaw[4:0],InputWattmeterReverseCountsRaw[4:0],OutputWattmeterForwardCountsRaw[4:0],OutputWattmeterReverseCountsRaw[4:0],PIF[4:0],PIR[4:0],POF[4:0],POR[4:0],*,checksum

Show Current Parameters Case statement (update Values)

1,2	IgSlope
3	VgAve Slope
4	Output Forward Power Offset
5	Out Pf Slope
6	Out Refleted power Offset
7	Out reflected power slope
8	input forward power offset
9	input forward ower slope
10	input reflected power offset
11	input reflected power slope
12	Frequency counter gate time
13	Cathode bias setting
$FBH, $FCH	Write to EEPROM

**end

Preamble	Sentence number	Data	Data
$APA	00	SerialNumber[0:11]	ESN(Hex)[0:4]+CH1+CH2
$APA	02	ForwardPower[4:0]	SWR (tents)[2:0]
$APA	03	Plus5(tenths)[2:0]	Plus12(tenths)[2:0]
$APA	04	loop 17 times generating edgeTable[4:0]	*
$APA	05	Band[1]+Segment[1]	Antenna (Hex)+"H"
$APA	06	Log Entries loop Byte(Hex)+CH1+CH2	Log Entries loop Flog(Hex)+CH1+CH2
$APA	07	Band	Segment 0 Freq[4:0]
$APA	08	Band	Segment
$APA	09	InVfRaw[3:0]	InVrRaw[3:0]
$APA	10	TempWord1(Hex)+CH1+CH2	TempWord0(Hex)+CH1+CH2
$APA	11	IgM[2:0]	IgB[2:0]
$APA	12	Band	InputWattmeterForwardPowerSlope[2:0]
$APA	13	Band	InputWattmeterForwardCountsRaw[4:0]

Data	**Data**	**Data**
Version(Hex)[0:4]+CH1+CH2	MainsBoardSWVer(Hex)[0:3]+CH1+CH2	DisplayControlerSWVer[0:3]
InputPower[3:0]	HighVoltage(volts)[3:0]	PlatCurrent(mA)[3:0]
Plus24(tenths)[2:0]	Minus12(tenths)[2:0]	Plus40(tenths)[2:0]
checksum		
LED_Dim+LED_Snd+LED_PEP+LED_Del+"H"+State	WarmupTime(Hex)+CH1+CH2	TuneValue(Hex)+CH1+CH2
*	checksum	
Segment 1 Freq[4:0]	Segment 2 Freq[4:0]	Segment 3 Freq[4:0]
TuneCapSetting[2:0]	LoadCapSetting[2:0]	FreqByte1(Hex)+CH1+CH2
OutVfRaw[3:0]	OutVrRaw[3:0]	VgRaw[3:0]
16 Eeprom Values(Hex)	CH1+CH2	*
VgM[2:0]	Grid[2:0]	I2C Baud Rate Divisor[2:0]
InputWattmeterInterceptSignBit	InputWattmeterForwardPowerIntercept[2:0]	InputWattmeterReversePowerSlope[2:0]
InputWattmeterReverseCountsRaw[4:0]	OutputWattmeterForwardCountsRaw[4:0]	OutputWattmeterReverseCountsRaw[4:0]

Data	**Data**	**Data**
StepperMotorSWVer[0:3]	SountGenSoftwareVer(Hex)[0:3]+CH1+CH2	*
Gain(tenths)[2:0]	GridVoltage(-1*tenths)[2:0]	GridCurrent(ma)[2:0]
ACLine(tenths)[3:0]	MainsBoardStatus[2:0]	MainsBoardTap
LoadValue(Hex)+CH1+CH2	DisplayedFault(HEX)+CH1+CH2	AutoTuneState
Segment 4 Freq[4:0]	*	checksum
FreqByte0(Hex)+CH1+CH2	FreqCounterValid	*
IgRaw[3:]	*	checksum
checksum		
*	checksum	
InputWattmeterInterceptSignBit	InputWattmeterReversePowerIntercept[2:0]	OutputWattmeterForwardPowerSlope[2:0]
PIF[4:0]	PIR[4:0]	POF[4:0]

Data	**Data**	**Data**
Checksum		
Band	AmpState	FaultCode[1:0]
TempSign[1]+Temp[2:0]+"."+TempDecimal	ExternalFanState	*
WasteHeat(watts)[3:0]	IgnoreBand	ShowHV/WasteHeat
checksum		
OutputWattmeterInterceptSignBit	OutputWattmeterForwardPowerIntercept[2:0]	OutputWattmeterReversePowerSlope[2:0]
POR[4:0]	*	checksum

Data	**Data**	**Data**	**Data**
Key Sense Checksum	ForwardPowerPEP[4:0]	*	Checksum
*	checksum		
OutputWattmeterInterceptSignBit	OutputWattmeterReversePowerIntercept[2:0]	*	checksum