

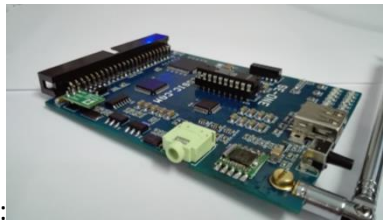


## SE-ONE

MSX MP3 Player, FM receiver, USB host



MSX:



CPC:

### Description:

The SE-ONE is an extension cartridge for the MSX and CPC home computers. You can play mp3 music files on your home computer.

The SE-ONE has a lot of modes. This mode and another option can be set and read with a user-friendly AT command set or Register commands.

Mode MP3A is the Sunrise compatible mode, there is IO compatibility with the Sunrise mp3 cartridge.

The FM mode turns the SE-ONE into a FM radio. European and a US Japan bands are available.

### Features:

- Fully user-friendly AT command set
- MP3 chip VS1053 decodes multiple formats:
- Mostly compatible with the Sunrise MP3 player
- FM stereo radio
- FM US/Europe (87.5 MHz to 108 MHz) and Japanese (76 MHz to 91 MHz) FM band.
- Onboard DSP Hi-fi stereo audio processor
- Stereo VU LED bar
- MSX input audio switch
- System updatable with a DFU cable and pc software
- Used IO ports for MSX : &h20 - &h0x27
- User IO ports for CPC: &FF20 - &FF27
- Connected pins : +5V, +12V, Busdir, Wait, Int.
- USB 2.0 host controller

### **Thanks to:**

EdoZ. (SymbOS),RJB (MSX / TEST),Totaly (webshop),Emil S (MSX hardware),Hans O (MSX hardware),MVM(MSX computerclub),Frits H.(C controllers),Kebu synthesizer music,Prodatron(SymbOS /CPC) ,GFlorez(Enterprise)

Also a word of thank to the Sunrise Foundation for the clear manual / documentation of the MP3 cartridge. This has contributed to the creation of the SE ONE, as it is now.

### **MODES:**

The SE-ONE has multiple modes:

*MP3A:* this mode is I/O identical to the SUNRISE MP3 player, there are differences in the software because the SE-ONE has another chipset and all of the functions convert to this chipset.

*MP3B:* mode: this mode you can play music files from USB stick

*FM:* The SE-ONE is an FM Radio

*REG:* Register functions , faster than AT commands

*TEST:* Test mode

### **USB:**

The USB host port can be use for USB stick (Mass storage device), for playing MP3 files.

Also with the USB port, you can update the Firmware with the included "DFU-cable" and software for the PC You can find the software and DFU-files for the SE-ONE on the website of [www.tmtlogic.com](http://www.tmtlogic.com) tab support

### ***CAUTION !!***

Use the "DFU cable" only for updating the SE-ONE. Not for anything else. This can cause serious damage.

Never use the USB portal to charge your phone or other power using devices !!

### **AT commandos:**

The protocol of the AT-command set has been used for a long time. AT-commands exists of a short string of text with added settings or parameters. You can also request data from the hardware with the AT-commands. You need a small program to use AT-commands that sends the string of text to the SE-ONE and processes the response of the SE-ONE. They are working on a machine language program that will

For quick reference help with AT commands, you can also use the /H after a command.

For example: AT+SEMODE/H

### **Safety points:**

Only insert the SE-ONE in the MSX while this is turned off and is voltage-free! !

This also applies to when you unplug the SE-ONE from the MSX.

Carefully move the antenna and keep a firm hold on the cartridge.

**Warranty:**

The warranty of the SE-ONE is 6 months, provided that it is used properly as written in this manual

Covered by the warranty is:

- Damage during transportation
- hardware faults that have appeared during production
- defective Dfu cable

Not covered by warranty:

- Breakdown of the antenna
- Defective aux out plug
- Defective USB connector
- Changes or attempted repairs to the device, or unauthorized modification of the circuitry
- Any other damage, including by improper use (e.g. placing or removing the SE-One while the MSX computer is turned on and power on the cartridge is locked)

**Liability:**

TMTlogic / Rinus and all other people who have cooperated on this project do not accept any liability for the damage that may have appeared during the use of the SE-ONE.

**IO Settings modes:**

	<b>MP3A</b>	<b>MP3B</b>	<b>REG</b>	<b>FM</b>	<b>TEST</b>	
Rd 0x20	At command	At command	At command	At command	At command	
Wr	At command	At command	At command	At command	At command	
Rd 0x21					0x21	
Wr					= Value	
Rd 0x22					0x22	
Wr					= Value	
Rd 0x23		Status byte *1	Status byte *1		0x23	
Wr					= Value	
Rd 0x24					0x24	
Wr					= Value	
Rd 0x25			Usb data		0x25	
Wr					= Value	
Rd 0x26		Vu left	Cmd respose		0x26	
Wr			Cmd Function		= Value	
Rd 0x27		Vu Right	Cmd Data		Value	
Wr			Cmd Data			

\*1 status byte

- bit 3 End of MUSIC playing
- bit 4 USB Error
- bit 5 USB busy

bit 6 Fifo buffer Full  
bit 7 Fifo buffer empty

## AT command reference:

---

---

### **CARTRIDGE**

---

---

#### **AT+CARTRIDGE**

SEONE/M	only MP3
SEONE/R	only Radio
SEONE/MR	MP3 and Radio

Get cartridge type

send: AT+CARTRIDGE?<13>  
response: AT+CARTRIDGE=SEONE/MR<13><10> options: SEONE/M,SEONE/R,SEONE/MR  
OK<13><10>

#### **AT+LIST**

Get the SE-ONE AT command's list

send: AT+LIST<13>  
response: AT+LIST=  
AT+CARTRIDGE<13>  
AT+SEMODE<13>  
AT+SELOGBOOK <13>  
.....text<13>  
.....text<13>  
OK<13><10> options: OK,ERR

#### **AT+SEMODE**

FM	= FM tuner
MP3A	= Sunrise MSXMP3
MP3B	= TMTLOGIC MP3
REG	= Register mode
TEST	= Test mode

Change mode from SE-ONE

send: AT+SEMODE=FM<13> options: FM,MP3A,MP3B,REG,TEST  
response: OK<13><10> options: OK,ERR,ILL

Get SE-ONE mode

send: AT+SEMODE?<13>  
response: AT+SEMODE=FM<13> options: FM,MP3A,MP3B,REG,TEST  
OK<13><10> options: OK,ERR,ILL

### **AT+SELOGBOOK**

Get SE-ONE system logbook

send: AT+SELOGBOOK?<13>  
response: AT+SELOGBOOK=TEXT<13> options: OK,ERR,ILL  
OK<13><10>

### **AT+SEVERSION**

- 1 = Radio
- 2 = MP3
- 3 = Radio and MP3

Get SE-ONE system SE-ONE version number

send: AT+SEVERSION?<13>  
response: AT+SEVERSION=1<13> options: 1,2,3  
OK<13><10> options: OK,ERR,ILL

### **AT+SEFIRMWARE**

Get firmware dfu file name

send: AT+SEFIRMWARE?<13>  
response: AT+SEFIRMWARE= SEONEddmmyyyy.DFU<13> options: SEONE[date].DFU  
OK<13><10> options: OK,ERR,ILL

### **AT+SETUNE**

Set startup tune

Set or Reset (ON/OFF) factory start-tune, this will be store in the ARM flash memory  
When set OFF, the SE-ONE play the factory start-tune not when system startup.

send: AT+SETUNE=ON<13> options: ON,OFF  
response: OK<13><10> options: OK,ERR,ILL

### **AT+SEPLAYTUNE**

Plays then intro tune from TMTLOGIC

send: AT+SEPLAYTUNE<13>  
response: OK<13><10> options: OK,ERR

### **AT+SEFIRMWARECHK**

Check if the firmware must be updated.

When firmware check is false, you must updating your SE-ONE !

send: AT+SEFIRMWARECHK=DDMMYYYY<13>  
response: AT+SEFIRMWARECHK=TRUE <13>  
OK<13><10> options: true, false  
options: OK,ERR,ILL

### **AT+SEFAULTNR**

Get the actual error number.

send: AT+SEFAULTNR ?<13>  
response: AT+ SEFAULTNR=12 <13><10> options: 0-29

0	"Succeeded	"
1	"Disk IO error	"
2	"Assertion failed	"
3	"The physical drive cannt work	"
4	"Could not find the file	"
5	"Could not find the path	"
6	"Invalid path name	"
7	"Access denied directory full	"
8	"Access denied	"
9	"Invalid file/directory	"
10	"Write protected	"
11	"Invalid logical drive	"
12	"The volume has no work area	"
13	"No valid FAT volume	"
14	"The f_mkfs() aborted	"
15	"Could not to access volume	"
16	"Rejected according sharing policy	"
17	"LFN could not be allocated	"
18	"Number of open files	"
19	"Invalid parameter	"

20	"Unknow at commando	"
21	"Syntax error	"
22	"Usb busy	"
23	"Invalid at command	"
24	"Usb already installed	"
25	"AT parameter error	"
26	"AT not in uppercase	"
27	"ARM flash error	"
28	"Invalid input	"
29	"Wrong SE mode	"

### **AT+SEFAULTTXT**

Get the actual error text.

send: AT+SEFAULTTXT? <13>  
 response: AT+ SEFAULTTXT= Invalid input <13><10>

---

### **FM Radio**

---

<i>AT+FMTYPE</i>	Get Radio chip type
<i>AT+FMFREQ</i>	Set FM radio frequentation
<i>AT+FMSCANUP</i>	Scan up to next radio channel
<i>AT+FMSCANDOWN</i>	Scan down to next radio channel
<i>AT+FMISM</i>	Change search mode on or off, its look alike the AFC function
<i>AT+FMmute</i>	Set FM radio mute
<i>AT+FMPLL</i>	Set FM PLL value
<i>AT+FMSSUD</i>	Set search up or down direction bit
<i>AT+FMSSL</i>	Set the search stop level
<i>AT+FMHLS</i>	Set the High Low Signal Injection bit
<i>AT+FMMS</i>	Change force stereo or mono
<i>AT+FMband</i>	Set the FM band range Europa and Japan/VS
<i>AT+FMHCC</i>	Set the High cut control bit
<i>AT+FMsnC</i>	Set the stereo noise canceling on/off
<i>AT+FMREADY</i>	Get the ready flag
<i>AT+FMBlF</i>	Get the band limit flag
<i>AT+FMSTEREO</i>	Get the stereo status from the FM channel
<i>AT+FMIF</i>	Get the intermediate frequency
<i>AT+FMADC</i>	Get the Analog signal level
<i>AT+FMSTARTFREQ</i>	Flashed the FM radio frequency in the ARM memory.

### **AT+FMTYPE**

Get Radio chip type

send: AT+FMTYPE?  
 response: AT+FMTYPE=TEA5767<13>                      options:TEA5767

OK<13><10>

options: OK,ERR,ILL

### **AT+FMFREQ**

Set FM radio frequention

send: AT+FMFREQ=88.7<13>

options: (76 to 108)\*

response: OK<13><10><0>

options: OK,ERR,ILL

Get FM Radio frequention

send: AT+FMFREQ?<13>

options: (76.0 to 108.9)\*

response: AT+FMFREQ=88.3<13>

OK<13><10><0>

options: OK,ERR,ILL

\* one decimal

### **AT+FMSCANUP**

Scan up to next radio channel

send: AT+FMSCANUP<13>

response: OK<13><10>

options: OK,ERR,ILL

### **AT+FMSCANDOWN**

Scan down to next radio channel

send: AT+FMSCANDOWN<13>

response: OK<13><10>

options: OK,ERR,ILL

### **AT+FMSM**

Change search mode on or off , its look alike the AFC function

send: AT+FMSM=ON<13>

options: ON,OFF

response: OK<13><10>

options: OK,ERR,ILL

Get the search mode

send: AT+FMSM><13>

response: AT+FMSM=ON<13>

OK<13><10>

options: ON,OFF

options: OK,ERR,ILL

bit 6 from byte 1 SM Search mode, if SM = 1= ON then in search mode; if SM = 0 = OFF then not in search mode



## **AT+FMMUTE**

Set FM radio mute

send: AT+FMMUTE=ON<13> options: ON,OFF  
response: OK<13><10> options: OK,ERR,ILL

Get FM radio mute state

send: AT+FMMUTE?<13>  
response: AT+FMSM=ON<13> options: ON,OFF  
OK<13><10> options: OK,ERR,ILL

bit 7 from byte 1 MUTE, if MUTE = 1 = ON then L and R audio are muted; if MUTE = 0 = OFF then L and R audio are not muted

## **AT+FMPLL**

Set FM PLL value

send: AT+FMPLL=1<13> options: (0-8191) 13 bits  
response: OK<13><10> options: OK,ERR,ILL

Get FM PLL value

send: AT+FMPLL?<13>  
response: AT+FMPLL=1<13> options: (0-8191) 13 bits  
OK<13><10> options: OK,ERR,ILL

5 to 0 PLL[13:8] + 7 to 0 PLL[7:0] from byte 1 and 2 setting of synthesizer programmable counter for search or preset

## **AT+FMSUD**

Set search up or down direction bit

send: AT+FMSUD=UP<13> options: UP,DOWN  
response: OK<13><10> options: OK,ERR,ILL

Get search direction

send: AT+FMSUD?<13>  
response: AT+FMFMSUD=DOWN<13> options: UP,DOWN  
OK<13><10> options: OK,ERR,ILL

bit 7 from byte 3 SUD Search Up/Down, if SUD = 1 = ON then search up; if SUD = 0 = OFF then search down

### **AT+FMSSL**

Set the search stop level

send: AT+FMSSL=1<13> options: 1,2,3  
response: OK<13><10> options: OK,ERR,ILL

Get the search stop level value

send: AT+FMSSL?<13>  
response: AT+FMSSL=1<13> options:1,2,3  
OK<13><10> options: OK,ERR,ILL

bit 6 and 5 from byte 3 SSL[1:0] Search Stop Level:  
SSL1 SSL0 Search stop level

- 0) 0 0 not allowed in search mode
- 1) 0 1 low; level ADC output = 5
- 2) 1 0 mid; level ADC output = 7
- 3) 1 1 high; level ADC output = 10

### **AT+FMHLSI**

Set the High Low Signal Injection bit

send: AT+FMHLSI=HLGH<13> options: HIGH,LOW  
response: OK<13><10> options: OK,ERR,ILL

Get the High Low Signal Injection bit

send: AT+FMHLSI?<13>  
response: AT+FMHLSI=LOW<13> options: HIGH,LOW  
OK<13><10> options: OK,ERR,ILL

bit 4 from byte 3 HLSI High/Low Side Injection: if HLSI = 1 = HIGH then high side LO injection; if HLSI = 0 = LOW then low side LO injection

### **AT+FMMS**

Change force stereo or mono

send: AT+FMMS=STEREO<13> options: STEREO,MONO  
response: OK<13><10> options: OK,ERR,ILL

Get the force value

send: AT+FMMS?<13>  
response: AT+FMMS=MONO<13> options: STEREO,MONO  
OK<13><10> options: OK,ERR,ILL

bit 3 from byte 3 MS Mono to Stereo: if MS = 1 = MONO then forced mono; if MS = 0 = STEREO then stereo  
ON

### **AT+FM BAND**

Set the FM band range Europa and Japan/VS

send: AT+FM BAND=EUR<13> options: EUR,JAP  
response: OK<13><10> options: OK,ERR,ILL

Get the FM band range Europa and Japan/VS

send: AT+FM BAND?<13>  
response: AT+FM BAND=EUR<13> options: EUR,JAP  
OK<13><10> options: OK,ERR,ILL

bit 5 from 4e byte BL Band Limits: if BL = 1 JAP then Japanese FM band; if BL = 0= EUR then  
US/Europe FM band

### **AT+FM HCC**

Set the High cut control bit

send: AT+FM HCC=ON<13> options ON,OFF  
response: OK<13><10> options: OK,ERR,ILL

Get the High cut control bit

send: AT+FM HCC?  
response: AT+FM HCC=ON<13> options: ON,OFF  
OK<13><10> options: OK,ERR,ILL

bit 2 from 4e byte HCC High Cut Control: if HCC = 1 = ON then high cut control is ON; if HCC = 0  
= OFF then high cut control is OFF

### **AT+FM SNC**

Set the stereo noise canceling on/off

send: AT+FM SNC=ON<13> options ON/OFF

response: OK<13><10> options: OK,ERR,ILL

Get the stereo noise cancellation setting

send: AT+FMSNC?  
response: AT+FMSNC=ON<13> options: ON,OFF  
OK<13><10> options: OK,ERR,ILL

bit 1 from byte 4 SNC Stereo Noise Cancelling: if SNC = 1 = ON then stereo noise cancellation is ON; if SNC = 0 = OFF then stereo noise cancellation is OFF

### **AT+FMREADY**

Get the ready flag

send: AT+FMREADY?<13>  
response: AT+FMREADY=ON<13> options: ON,OFF  
OK<13><10> options: OK,ERR,ILL

bit 7 from 1e byte RF Ready Flag: if RF = 1 =ON then a station has been found or the band limit has been reached; if RF = 0 = OFF then no station has been found

### **AT+FMBLF**

Get the band limit flag

send: AT+FMBLF?<13>  
response: AT+FMBLR=ON<13> options: ON,OFF  
OK<13><10> options: OK,ERR,ILL

bit 6 from 1e byte BLF Band Limit Flag: if BLF = 1 = ON then the band limit has been reached; if BLF = 0 =OFF then the band limit has not been reached

### **AT+FMSTEREO**

Get the stereo status from the FM channel

send: AT+STEREO?<13>  
response: AT+FMSTEREO=ON<13> options: ON,OFF  
OK<13><10> options: OK,ERR,ILL

bit 7 byte 2 STEREO Stereo indication: if STEREO = 1 = ON then stereo reception; if STEREO = 0 = OFF then mono reception

### **AT+FMIF**

Get the intermediate frequency

send: AT+FMIF?<13>  
response: AT+FMIF=123<13> options:0-127 7 bits  
OK<13><10> options: OK,ERR,ILL

IF[7:0] IF counter result

### **AT+FMADC**

Get the Analog signal level

send: AT+FMADC?<13>  
response: AT+FMADC=5<13> options: 0-15 4 bit  
OK<13><10> options: OK,ERR,ILL

bit 7 to 4 from 4e byte level ADC output

### **AT+FMSTARTFREQ**

Flashed the FM radio frequency in the ARM memory.  
When the radio turns on, the radio starts with this frequency  
the Set FM radio frequency

send: AT+FMSTARTFREQ=88.7<13> options: (76 to 108)\*  
response: OK<13><10> options: OK,ERR,ILL

Get FM Radio frequency

send: AT+FMSTARTFREQ?<13> options: (76 tot 108)\*  
response: AT+FMFREQ=88.7<13>  
OK<13><10> options: OK,ERR,ILL

\* one decimal

---

## **DSP Digital Sound Processor / Audio Processor**

---

<i>AT+DSPTYPE</i>	Get the DSP chip type
<i>AT+DSPVOLL</i>	Set the left volume level
<i>AT+DSPVOLR</i>	Set the right volume level
<i>AT+DSPBASS</i>	Set the bass level
<i>AT+DSPTREBLE</i>	Set the treble volume level
<i>AT+DSPVOLUP</i>	Increase the volume by 10 steps
<i>AT+DSPVOLDOWN</i>	Decrease the volume by 10 steps
<i>AT+DSPMUTE</i>	Set DSP mute

### **AT+DSPTYPE**

Get the DSP chip type

send: AT+DSPTYPE?<13>  
response: AT+DSPTYPE=TDA8425<13>  
OK<13><10>

options: TDA8425  
options: OK,ERR,ILL

### **AT+DSPVOLL**

Set the left volume level

send: AT+DSPVOLL=80<13>  
response: OK<13><10>

options: 0-100  
options: OK,ERR,ILL

Get the left volume level

send: AT+DSPVOLL?<13>  
response: AT+DSPVOLL=66<13>  
OK<13><10>

options: 0-100  
options: OK,ERR,ILL

### **AT+DSPVOLR**

Set the right volume level

send: AT+DSPVOLR=80<13>  
response: OK<13><10>

options: 0-100  
options: OK,ERR,ILL

Get the left volume level

send: AT+DSPVOLR?<13>  
response: AT+DSPVOLR=66<13>  
OK<13><10>

options: 0-100  
options: OK,ERR,ILL

### **AT+DSPBASS**

Set the bass level

send: AT+DSPBASS=80<13>  
response: OK<13><10>

options: 0-100  
options: OK,ERR,ILL

Get the bass level

send: AT+DSPBASS?<13>  
response: AT+DSPBASS=66<13>  
OK<13><10>

options: 0-100  
options: OK,ERR,ILL

### **AT+DSPTREBLE**

Set the treble volume level

send: AT+DSPTREBLE=80<13> options: 0-100  
response: OK<13><10> options: OK,ERR,ILL

Get the left volume level

send: AT+DSPTREBLE?<13>  
response: AT+DSPTREBLE=66<13> options: 0-100  
OK<13><10> options: OK,ERR,ILL

### **AT+DSPVOLUP**

Increase the volume by 10 steps

send: AT+DSPVOLUP<13>  
response: OK<13><10> options: OK,ERR

### **AT+DSPVOLDOWN**

Decrease the volume by 10 steps

send: AT+DSPVOLDOWN<13>  
response: OK<13><10> options: OK,ERR

### **AT+DSPMUTE**

Set DSP mute

send: AT+DSPMUTE=ON<13> options: ON,OFF  
response: OK<13><10> options: OK,ERR,ILL

Get DSP mute state

send: AT+DSPMUTE?<13>  
response: AT+DSPMUTE=ON<13> options: ON,OFF  
OK<13><10> options: OK,ERR,ILL

---

### **MP3 DSP chip**

**AT+MP3TYPE** Get the mp3 chip type  
**AT+MP3LOWLEVEL** Set the low-level bit from status register inp(&h23) bit 7  
**AT+MP3HIGHLEVEL** Set the high-level bit from status register inp(&h23) bit 6

### **AT+MP3TYPE**

Get the mp3 chip type

send: AT+MP3TYPE?<13>  
response: AT+ MP3TYPE=VS1053<13> options: VS1053  
OK<13><10>

### **AT+MP3LOWLEVEL**

Set the low-level bit from status register inp(&h23) bit 7

Note: this works only in MP3B mode !

When the content of the audiostream fifo lower is than this value, bit 7 will be set on

send: AT+MP3LOWLEVEL= 32<13> options: 0-65535  
response: OK<13><10> options: OK,ERR,ILL

### **AT+MP3HIGHLEVEL**

Set the high-level bit from status register inp(&h23) bit 6

Note: this works only in MP3B mode !

When the content of the audiostream fifo higher is than this value, bit 6 will be set on

send: AT+MP3HIGHLEVEL= 12232<13> options: 0-65535  
response: OK<13><10> options: OK,ERR,ILL

---

### **VU meter**

---

*AT+VULEFT* Set the left VU byte  
*AT+VURIGHT* Get the left VU byte  
*AT+VULN* Get the left VU number  
*AT+VURN* Get the right VU number

*rserve at+vutype*

### **AT+VULEFT**

Set the left VU byte

send: AT+VULEFT=80<13> options: 0-255  
response: OK<13><10> options: OK,ERR,ILL



Get the left VU byte

send: AT+VULEFT?<13>  
response: AT+VULEFT=66<13> options: 0-255  
OK<13><10> options: OK,ERR,ILL

### **AT+VURIGHT**

Get the left VU byte

send: AT+VURIGHT=80<13> options: 0-255  
response: OK<13><10> options: OK,ERR,ILL

Get the left VU byte

send: AT+VURIGHT?<13>  
response: AT+VURIGHT=66<13> options: 0-255  
OK<13><10> options: OK,ERR,ILL

### **AT+VULN**

Get the left VU number

send: AT+VULN?<13>  
response: AT+VULN=7<13> options: 0-8  
OK<13><10> options: OK,ERR,ILL

Get IO base: PRINT INP(&H26)

### **AT+VURN**

Get the right VU number

send: AT+VURN?<13>  
response: AT+VURN=7<13> options: 0-8  
OK<13><10> options: OK,ERR,ILL

Get IO base: PRINT INP(&H27)

---

## **MSX**

**AT+MSXAUDIO** Set MSX aux in switch  
**AT+MSXSTARTAUDIO** Flashed the this settings in the ARM memory.

## **AT+MSXAUDIO**

Set MSX aux in switch

Signal is the left output channel to pin 49 if the msx connector

send: AT+MSXAUDIO=ON<13> options: ON,OFF  
response: OK<13><10> options: OK,ERR,ILL

Get MSX aux line status

send: AT+MSXAUDIO?<13>  
response: AT+MSXAUDIO=OFF<13> options: ON,OFF  
OK<13><10> options: OK,ERR,ILL

## **AT+MSXSTARTAUDIO**

Flashed the this settings in the ARM memory. When the SE-ONE turns on, this setting will be used

Set MSX aux in switch

Signal is the left output channel

send: AT+MSXSTARTAUDIO=ON<13> options: ON,OFF  
response: OK<13><10> options: OK,ERR,ILL

Get MSX aux line status

send: AT+MSXSTARTAUDIO?<13>  
response: AT+MSXSTARTAUDIO=OFF<13> options: ON,OFF  
OK<13><10> options: OK,ERR,ILL

---

## **USB universal serial bus**

---

**AT+USBINIT** Install usb port

**AT+USBDEVICE** Get which USB device is connected

### **AT+USBINIT**

Note: only in MP3B mode

Install usb port

When USB is good installed, the blue led is burn on the SE-ONE

send: AT+USBINIT<13>  
response: OK<13><10> options: OK,ERR,ILL

## **AT+USBDEVICE**

Note: only in MP3B mode

NONE: no device

MSD: Mass storage device (usb stick)

Get which USB device is connected

send: AT+USBDEVICE?<13>  
response: AT+USBDEVICE =MSD<13> options: NONE,MSD  
OK<13><10> options: OK,ERR,ILL

---

### **MSD mass storage device (only MP3B & REG mode)**

---

**AT+MSDPATH** Set the directory from the Mass Storage Device (usb stick)  
**AT+MSDFILES?** Get the available files and directories in the current directory.  
**AT+MSDPLAY** Play an music file from Mass Storage Device (usb stick)  
**AT+MSDPAUSE** Pause playing  
**AT+MSDSTART** Continue music playing  
**AT+MSDSTOP** Stop playing  
**AT+MSDTALK** Play a speech sample from usb /setalk

## **AT+MSDPATH**

Note: only in MP3B mode

Set the directory from the Mass Storage Device (usb stick)

send: AT+MSDPATH=/MAP/MAP<13>  
response: OK<13><10> options: OK,ERR,ILL

Get the directory from the Mass Storage Device (usb stick)

send: AT+MSDPATH?<13>  
response: AT+MSDPATH=/MAP/MAP<13>  
OK<13><10> options: OK,ERR,ILL

## **AT+MSDFILES?**

Note: only in MP3B mode

Get the available files and directories in the current directory.

send: AT+MSDFILES?<13>  
response: OK<13><10> options: OK,ERR,ILL

for read the files run this program:

```
1000 D = INP(&H25)
1010 PRINT CHR$(D);
1020 IF D = 10 THEN END
1030 GOTO 1000
```

### ***AT+MSDPLAY***

Note: only in MP3B mode  
This function works stand-alone in the SE\_ONE

Play an music file from Mass Storage Device (usb stick)

send: AT+MSDPLAY=MUSIC.MP3<13>  
response: OK<13><10> options: OK,ERR,ILL

### ***AT+MSDPAUSE***

Note: only in MP3B mode

Pause playing

send: AT+MSDPAUSE<13>  
response: OK<13><10> options: OK,ERR,ILL

### ***AT+MSDSTART***

Note: only in MP3B mode

Continue music playing

send: AT+MSDSTART<13>  
response: OK<13><10> options: OK,ERR,ILL

### ***AT+MSDSTOP***

Note: only in MP3B mode

Stop playing

send: AT+MSDSTOP<13>  
response: OK<13><10> options: OK,ERR,ILL

### ***AT+MSDTALK***

Note: only in MP3B and REG mode  
Play app switch to streaming mode.  
Speech samples can be download from [www.tmtlogic.com/support](http://www.tmtlogic.com/support)  
Place the samples in \SETALK directory

Play a speech sample from usb /setalk

send: AT+MSDTALK=44<13> 0-57 \*  
response: OK<13><10> options: OK,ERR,ILL

wait for usb 140 if (inp(&h23) AND 32) = 32 ) then goto 140 bit 5 USB\_busy

see data sheet of the SP0256

<http://www.futurebots.com/spo256.pdf>

hex	??oct	word	time
00	000 PA1	PAUSE	10MS
01	001 PA2	PAUSE	30MS
02	002 PA3	PAUSE	50MS
03	003 PA4	PAUSE	100MS
04	004 PA5	PAUSE	200MS
05	005 /OY/	BOY	420MS
06	006 /AY/	Sky	260MS
07	007 /EH/	End	70MS
08	010 /KK3/	Comb	120MS
09	011 /PP/	Pow	210MS
0A	012 /JH/	Dodge	140MS
0B	013 /NN1/	Thin	140MS
0C	014 /IH/	Sit	70MS
0D	015 /TT2/	To	140MS
0E	016 /RR1/	Rural	170MS
0F	017 /AX/	Succeed	70MS
10	020 /MM/	Milk	180MS
11	021 /TT1/	Part	100MS
12	022 /DH1/	They	290MS
13	023 /IY/	See	250MS
14	024 /EY/	Beige	280MS
15	025 /DD1/	Could	70MS
16	026 /UW1/	To	100MS
17	027 /AO/	Aught	100MS
18	030 /AA/	Hot	100MS
19	031 /YY2/	Yes	180MS
1A	032 /AE/	Hat	120MS
1B	033 /HH1/	He	130MS
1C	034 /BB1/	Business	80MS
1D	035 /TH/	Thin	180MS
1E	036 /UH/	Book	100MS

1F 037	/UW2/	Food	260MS
20 040	/AW/	Out	370MS
21 041	/DD2/	Do	160MS
22 042	/GG3/	Wig	140MS
23 043	/VV/	Vest	190MS
24 044	/GG1/	Got	80MS
25 045	/SH/	Ship	160MS
26 046	/ZH/	Azure	190MS
27 047	/RR2/	Brain	120MS
28 050	/FF/	Food	150MS
29 051	/KK2/	Sky	190MS
2A 052	/KK1/	Can't	160MS
2B 053	/ZZ/	Zoo	210MS
2C 054	/NG/	Anchor	220MS
2D 055	/LL/	Lake	110MS
2E 056	/WW/	Wool	180MS
2F 057	/XR/	Repair	360MS
30 060	/WH/	Whig	200MS
31 061	/YY1/	Yes	130MS
32 062	/CH/	Church	190MS
33 063	/ER1/	Fir	160MS
34 064	/ER2/	Fir	300MS
35 065	/OW/	Beau	240MS
36 066	/DH2/	They	240MS
37 067	/SS/	Vest	90MS
38 070	/NN2/	No	190MS
39 071	/HH2/	Hoe	180MS
3A 072	/OR/	Store	330MS
3B 073	/AR/	Alarm	290MS
3C 074	/YR/	Clear	350MS
3D 075	/GG2/	Guest	40MS
3E 076	/EL/	Saddle	190MS
3F 077	/BB2/	Business	50MS

#### AT+MSDSETSAMPLE

Note: only in MP3B and REG mode  
 Store sample filename in memory 0-9.  
 Place the samples in \SAMPLE directory

Play a speech sample from usb /setalk

send: AT+MSDSETSAMPLE=1@TOETER.MP3<13>  
 response: OK<13><10>

options: OK,ERR,ILL

#### AT+MSDSAMPLE

Note: only in MP3B and REG mode  
Play app switch to streaming mode.  
Play sample from USB/SAMPLE  
Place the samples in \SAMPLE directory

Play a speech sample from usb /setalk

send: AT+MSDSAMPLE=1<13> 0-9  
response: OK<13><10> options: OK,ERR,ILL

---

### **REG register mode (beta)**

---

Before test this mode run this command :AT+SEMODE=REG

(AT commands are active in REG mode)

I/O port setting see begin of this manual

CALL ARM response

```
1000 I = INP(&h26)
1010 IF I = 1 THEN GOTO 1010
1020 IF I = 2 THEN PRINT "Error" : end
1030 RETURN
```

---

### **Audio DSP**

---

F16	cmd Set Dsp Select	17 sept 2018
F17	cmd Get Dsp Select	7 sept 2018
F48	cmdDspVolUp	17 sept 2018
F49	cmdDspVolDown	17 sept 2018
F50	cmdSetVolLeftRight	17 sept 2018
F51	cmdGetVolLeftRightStereo	18 sept 2018
F52	cmdSetBassTreableStereo	18 sept 2018
F53	cmdGetBassTreableStereoMute	18 sept 2018

**F16** **cmd Set Dsp Select** 17 sept 2018

Select source speaker  
0 = FM chip  
1 = MP3 chip

example select sound from MP3 chip

100 out &h26,0

110 out &h27,1

120 out &h26,16

130 gosub ARM response

Start wr pointer

1 = MP3 chip

Parameters when set FM:

```
mute = TEA5767_MUTE_OFF;
sm    = TEA5767_SEARCHMODE_OFF;
pll   = 0;
sud   = TEA5767_SUD_UP;
ssl   = TEA5767_SCAN_STOP_MID;
hilo  = TEA5767_HILO_HIGH;
ms    = TEA5767_MONO;
bl    = TEA5767_EUROPE_BAND;
hcc   = TEA5767_HCC_ON;
snc   = TEA5767_SNC_OFF;
```

**F17 cmd Get Dsp Select**

7 sept 2018

0 = FM chip

1 = MP3 chip

100 out &h26,17

130 gosub ARM response

140 print inp(&h27)

activate function

**F48 cmdDspVolUp**

17 sept 2018

100 out &h26,48

130 gosub ARM response

activate function

**F49 cmdDspVolDown**

17 sept 2018

100 out &h26,49

130 gosub ARM response

activate function

**F50 cmdSetVolLeftRight**

17 sept 2018

value 0-100d

100 out &h26,0

110 out &h27,45      volume left



120 out &h27,55      volume Right  
130 out &h26,50  
140 gosub ARM response

**F51**      **cmdGetVolLeftRightStereo**      18 sept 2018

100 out &h26,51  
110 gosub ARM response

120 print inp(&h27)  
130 print inp(&h27)

**F52**      **cmdSetBassTreableStereo**      18 sept 2018

bass,treable      value 0-100d  
stereo type      value 0-3

0 =      Fored mono  
1 =      linear Stereo  
2 =      pseudo Stereo  
3 =      spatial Stereo

mute              value 0-1

100 out &h26,0  
110 out &h27,45      bass  
120 out &h27,55      treable  
130 out &h27,2      Stereo type  
140 out &h27,1 Mute  
150 out &h26,52  
160 gosub ARM response

**F53**      **cmdGetBassTreableStereoMute**      18 sept 2018

100 out &h26,53  
110 gosub ARM response  
120 print inp(&h27)  
130 print inp(&h27)  
140 print inp(&h27)  
150 print inp(&h27)

---

**FM:**

---

F32	cmd Fm Wr Registers to tea5767	16 sept 2018
F33	cmd Fm rd registers from tea5767	17 sept 2018
F34	cmd Fm Set Freq (ASCII)	17 sept 2018
F35	cmd Fm Get Freq (ASCII)	17 sept 2018
F36	cmd Fm Set Freq (binair)	17 sept 2018
F37	cmdFmGetBinFreq	17 sept 2018

F38	cmdFmScanUpDown	18 sept 2018
<b>F32</b>	<b>cmd Fm Wr Registers to tea5767</b>	16 sept 2018
	100 out &h26,0	
	110 out &h27,data register 0	
	120 out &h27,data register 1	
	130 out &h27,data register 2	
	140 out &h27,data register 3	
	150 out &h27,data register 4	
	160 out &h26,32	activate function
	170 gosub ARM response	
<b>F33</b>	<b>cmd Fm rd registers from tea5767</b>	17 sept 2018
	100 out &h26,33	activate function
	110 gosub ARM response	
	120 print inp(&h27)	reg 0
	130 print inp(&h27)	reg 1
	140 print inp(&h27)	reg 2
	150 print inp(&h27)	reg 3
	160 print inp(&h27)	reg 4
<b>F34</b>	<b>cmd Fm Set Freq (ASCII)</b>	17 sept 2018
	100 out &h26,0	
	110 out &h27,'8'	
	120 out &h27,'8'	
	130 out &h27,'.'	
	140 out &h27,'0'	
	150 out &h27,'0'	
	160 out &h26,34	activate function
	170 gosub ARM response	
<b>F35</b>	<b>cmd Fm Get Freq (ASCII)</b>	17 sept 2018
	100 out &h26,35	activate function
	110 gosub ARM response	
	120 print chr\$(inp(&h27))	'1'
	130 print chr\$(inp(&h27))	'0'
	140 print chr\$(inp(&h27))	'0'
	150 print chr\$(inp(&h27))	'.'
	160 print chr\$(inp(&h27))	'4'
	170 print inp(&h27)	<13>



0 = OFF  
1 = ON

100 out &h26,0  
110 out &h27,1 = ON  
120 out &h26,64

130 gosub ARM response

**F66** *cmdSetVuLeftRight* 18 sept 2018

100 out &h26,0  
110 out &h27,1 left byte  
120 out &h27,1 right byte  
130 out &h26,66  
140 gosub ARM response

**F67** *cmdGetVuLeftRight* 18 sept 2018

100 out &h26,67  
110 gosub ARM response  
120 print inp(&h27) Vu left byte leds  
130 print inp(&h27) Vu right byte leds

**F68** *cmdSetVuMaxLeds* 18 sept 2018

value is 1-32

100 out &h26,0  
110 out &h27,8 Default 8  
120 out &h26,68  
130 gosub ARM response

**F69** *cmdGetVuLeftRightNumber* 18 sept 2018

is the same as in MP3B inp(&h26) & inp(&h27)

value is: 0 - 8

100 out &h26,69  
110 gosub ARM response  
120 print inp(&h27) Vu left n leds  
130 print inp(&h27) Vu right n leds

**F70** *cmdGetEqualizer* (todo) 19 sept 2018

Not for FM radio sound

100 out &h26,&h46  
110 gosub ARM response

function cmdGetEqualizer

todo 16x print inp(&h27)  
print inp(&h27) Vu right n leds

=====  
FAT32  
reserve 50-5F

case 50: addsCmdInitFatfs();  
case 51: addsCmdReadSector();  
case 52: addsCmdWriteLock();  
case 53: addsCmdWriteSector();  
case 54: addsCmdFatfsInfo();

=====  
SYSTEM:

60h cmdGetFirmwareVersion 18 sept 2018

Attention! another structure then AT+SEFIRMWARE !

YEAR  
YEAR  
YEAR  
YEAR  
MOUNT  
MOUNT  
DAY  
DAY

out &h26,&h60

function cmdGetFirmwareVersion

inp(&h26)  
wait for ilde..  
while(inp(&h26) == 1);

0=idle, 1= busy, 2=Error

print chr\$(inp(&h27)) 2  
print chr\$(inp(&h27)) 0  
print chr\$(inp(&h27)) 1  
print chr\$(inp(&h27)) 8  
print chr\$(inp(&h27)) 0  
print chr\$(inp(&h27)) 9  
print chr\$(inp(&h27)) 1  
print chr\$(inp(&h27)) 8

```
print inp(&h27)      <0>
```

#### BASIC example Firmware check

```
10 OUT &H26,&h60
20 IF INP(&H26) = 1 THEN GOTO 20
21 IF INP(&H26) = 2 THEN PRINT"ERROR":END
30 F$ = ""
40 I = INP(&H27)
50 IF I = 0 THEN GOTO 80
60 F$=F$+CHR$(I)
70 GOTO 40
80 C = 20170909   :rem Year Mount Day
90 F = VAL(F$)
99 IF F < C THEN PRINT"Update your SE-ONE" :END
```

=====

#### USB MSD:

```
70h   cmdUsbMsdInit           18 sept 2018

      out &h26,&h70           function      cmdUsbMsdInit

      inp(&h26)               0=idle, 1= busy, 2=Error
      wait for ilde..
      while(inp(&h26) == 1);

72h   cmdUsbMsdAvailable

74h   cmdSetMSDPLAY          19 sept 2018

      out &h26,0
      out &h27,asc("F")
      out &h27,asc("I")
      out &h27,asc("L")
      out &h27,asc("E")
      out &h27,asc("N")
      out &h27,asc("A")
      out &h27,asc("M")
      out &h27,asc("E")
      out &h27,asc(".")
      out &h27,asc("B")
      out &h27,asc("A")
      out &h27,asc("S")
```



```

print chr$(inp(&h27)) \
print chr$(inp(&h27)) M
print chr$(inp(&h27)) A
print chr$(inp(&h27)) P
print inp(&h27)      <0>

```

78h cmdMsdStart 19 sept 2018

out &h26,&h78 Function cmdMsdStart

inp(&h26) 0=idle, 1= busy, 2=Error  
wait for ilde..  
while(inp(&h26) == 1);

79h cmdMsdStop 19 sept 2018

out &h26,&h79 Function cmdMsdStop

inp(&h26) 0=idle, 1= busy, 2=Error  
wait for ilde..  
while(inp(&h26) == 1);

7Ah cmdMSDPAUSE 19 sept 2018

out &h26,&h79 Function cmdMsdPause

inp(&h26) 0=idle, 1= busy, 2=Error  
wait for ilde..  
while(inp(&h26) == 1);

7Bh cmdMSDFILES 19 sept 2018

8 characters + ' . ' + 3 characters + <TAB>  
out &h26,&h7B Function cmdMsdFiles

inp(&h26) 0=idle, 1= busy, 2=Error  
wait for ilde..  
while(inp(&h26) == 1);

```

10 i = inp(&h25)          note: 25h !
20 if i = 0 then end
30 print chr#(i);
40 goto 10

```

file , m p 3 <32> <32> <32> <32> <9> file n



7Ch cmdGetMSDSIZE 19 sept 2018

Gives the size of the file after play commando

32 bits binair value

out &h26,&h77 function cmdGetMSDPATH

inp(&h26) 0=idle, 1= busy, 2=Error

wait for ilde..

while(inp(&h26) == 1);

```
print chr$(inp(&h27)) \
print chr$(inp(&h27)) M
print chr$(inp(&h27)) A
print chr$(inp(&h27)) P
print inp(&h27) <0>
```

7Dh cmdGetMSDCNT 19 sept 2018

value 0 - 100 %

out &h26,&h7D function cmdGetMSDCNT

inp(&h26) 0=idle, 1= busy, 2=Error

wait for ilde..

while(inp(&h26) == 1);

print inp(&h27)

### **F126 cmdSetMSDTALK**

note: Emulation of the SP0256 speech chip  
place in the directory USB:/SETALK/SP...H.MP3 files  
this can be download from [http://www.tmtlogic.com/SE\\_ONE/SETALK.ZIP](http://www.tmtlogic.com/SE_ONE/SETALK.ZIP)

more information see **AT+MSDTALK**

value 0 - 58

100 out &h26,0

110 out &h27,value

120 out &h26,126

130 if inp(&h26) = 1 then goto 130

140 if (inp(&h23) AND 32) = 32 ) then goto 140

function cmdGetMSDCNT

0=idle, 1= busy, 2=Error

bit 5 USB\_busy

**F127 cmdSetMSDSETSAMPLE**

note: place in the directory USB:/SAMPLE

value 0 - 9

filename max 12 characters ( filename.mp3 = 12 characters )

100 out &h26,0

110 out &h27,value

120 out &h27,asc("B")

130 out &h27,asc("E")

140 out &h27,asc("L")

150 out &h27,asc("L")

1620 out &h27,asc(".")

170 out &h27,asc("M")

180 out &h27,asc("P")

190 out &h27,asc("3")

200 out &h26,127

210 gosub ARM response

**F128 cmdSetMSDSAMPLE**

note: Play samples

place in the directory USB:/SAMPLE

value 0 - 9

100 out &h26,0

110 out &h27,value

120 out &h26,128

130 gosub ARM response

---

**Examples****How to use AT commands:**

AT.BAS

100 WIDTH 80

110 '

120 'Example: AT COMMAND:? AT+SEMODE?

130 '

140 'set caps lock

150 DEFUSR1=&HF36:A=USR1(0)

160 '

170 '

```

180 AT$=""
190 INPUT"AT COMMAND:";AT$
200 L = LEN(AT$)
210 IF L = 0 THEN GOTO 180
220 FOR T = 1 TO L
230 S = ASC(RIGHT$(LEFT$(AT$,T),1))
240 OUT &H20,S
250 NEXT T
260 OUT &H20,13
270 '
280 'read response
290 IF INKEY$<>"" THEN GOTO 180
300 '
310 I = INP(&H20)
320 IF I = 0 THEN GOTO 280
340 IF I = 13 THEN PRINT
350 IF I = 10 THEN GOTO 180
360 GOTO 280

```

### ***How to use the radio receiver?***

Use at.bas

```

AT+SEMODE=FM
AT+FMFREQ=88.0

```

```

For search UP:      AT+FMSCANUP
For search Down    AT+FMSCANDOWN

```

### ***How to play form USB stick?***

```

AT+SEMODE=MP3B
AT+USBINIT          The blue led must burn, than the USB port is active
AT+MSDPATH=/MAP     Set the directory , default is the root
AT+MSDPLAY=YOURFILE.MP3

```

```

For pause playing:  AT+MSDPAUSE
For continu playing: AT+MSDSTART
For stop playing:   AT+MSDSTOP

```

### ***How to get the files list form USB stick?***

```

AT+SEMODE=MP3B
AT+USBINIT          The blue led must burn, than the USB port is active
AT+MSDFILES?

```

BASIC:

for read the files run this program:

```
1000 D = INP(&H25)
1010 PRINT CHR$(D);
1020 IF D = 10 THEN END
1030 GOTO 1000
```

---

### ***Troubleshooting***

---

#### ***No Sound***

```
AT+MSXAUDIO=ON
AT+DSPVOLL=100
AT+DSPVOLR=100
```

#### ***USB don't work***

After mode MP3B and USBINIT the blue led must be burn on, otherwise reset the SE-ONE and try again.  
This is an bug in the mp3b software.

The usb stick is not good formatted, format it with FAT or FAT32