

EV6550DHAT Device Control API Reference

Version 1.0

7th November 2019

Introduction

This document describes an application programming interface (API) for controlling the CMX655D Voice Codec IC. The current implementation has been written to use either the I2C port of a Raspberry Pi2, 3 or 4.

Programmes that use this API should include the «stdgeo.h» and «655spt.h» header files and should include the «655spt.c» source file in the compilation.

example.c

```
#include "stdgeo.h"
#include "655spt.h"
int main(argc,argv)
int  argc;
char *argv[];
{
    ...
    if(CMX655Open()<0)
    {        fprintf(stderr,"Can't access the serial port\n");
        exit(1);
    }
    if(CMX655Command(CMD_CLOCK_START);
    ...
    CMX655Close();
    exit(0);
}
```

```
Pi>gcc -Wall -o example example.c 655spt.c
```

```
Pi>
```

For the I2C port, the device address is assumed to be 0x54. The 655spt.h header file defines «655» to be 0x54. If a different address is to be used, then this constant must be changed accordingly and all the software re-built.

For the I2C port the device name is assumed to be «/dev/i2c-1». The device name string can be found in the 655spt.c source code.

Note: The API list is not exhaustive but is sufficient for use with the EV6550DHAT and its associated GUI.

For additional functions and CMX655D Register designations please refer to the CMX655D datasheet.

I2cRead

Summary

```
#include "655spt.h"

int      I2cRead(fdPort,iSlave,iAddr,iCount,pBytes)

int      fdPort;
int      iSlave;
int      iAddr;
int      iCount;
PBYTE    pBytes;
```

Description

Read a number of bytes from a device connected to an I2C port.

fdPort specifies the file handle of the open I2C port.

iSlave specifies the address of the device on the I2C bus.

iAddr specifies the address of the register to start reading from.

iCount specifies the number of bytes to read.

pBytes points to the buffer where the bytes should be stored.

Returns

Returns -1 on error

I2cWrite

Summary

```
#include "655spt.h"

int      I2cWrite (fdPort, iSlave, iAddr, iCount, pBytes)

int      fdPort;
int      iSlave;
int      iAddr;
int      iCount;
PBYTE    pBytes;
```

Description

Write a number of bytes to a device connected to an I2C port.

fdPort specifies the file handle of the open I2C port.

iSlave specifies the address of the device on the I2C bus.

iAddr specifies the address of the register to start writing to.

iCount specifies the number of bytes to write.

pBytes points to the buffer containing the bytes to write.

Returns

Returns -1 on an error.

CMX655Clock

Summary

```
#include "655spt.h"

int      CMX655Clock(iOn)

int      iOn;
```

Description

Writes to the COMMAND (\$33) register, enables or disables the main clock and tests its status.

iOn toggles the main clock

1 = enables and blocks until clock ready or timed out.

<>1 to disable main clock

Returns

Returns -1 on error or time out.

CMX655Close

Summary

```
#include "655spt.h"

void      CMX655Close(void)
```

Description

Closes the previously open serial port used for communicating with the CMX655 device registers.

This function takes no arguments

Returns

This is a void function and therefore returns nothing.

CMX655ClrBitsSysCtrl

Summary

```
#include "655spt.h"

int      CMX655ClrBitsSysCtrl (iBits)

int      iBits;
```

Description

Clears selected bits in the SYSCTRL (\$32) register without affecting remaining bits, achieved by a read/write process.

iBits specifies the bits to be cleared. A 1 in any bit position will clear the associated register bit.

Returns

Returns -1 on error.

CMX655Command

Summary

```
#include "655spt.h"

int      CMX655Command(iCmd)

int      iCmd;
```

Description

Writes a command to the COMMAND (\$33) register.

iCmd specifies the command to write. The following values may be used:-

```
CMD_CLOCK_STOP
CMD_CLOCK_START
CMD_SOFT_RESET
```

Returns

Returns -1 on an error

CMX655GetAlcAtten

Summary

```
#include "655spt.h"

int      CMX655GetAlcAtten(void)
```

Description

Reads the current ALC attenuation value from the ALCSTAT (\$2E) register.

This function takes no arguments.

Returns

Returns the ALC attenuation which is a value between 0 (0dB) and 31 (31dB) inclusive.

Returns -1 on an error.

CMX655GetClkSrc

Summary

```
#include "655spt.h"

int      CMX655GetClkSrc(void)
```

Description

Reads the current setup for the internal clock source from the CLKCTRL (\$03) register.

This function takes no arguments.

Returns

Returns CLKCTRL_PLL_RCLK, CLKCTRL_PLL_LRCLK, CLKCTRL_LPO, CLKCTRL_RCLK bits.

Returns -1 on an error.

CMX655GetCpi

Summary

```
#include "655spt.h"

int      CMX655GetCPI(void)
```

Description

Reads the charge pump current gain setting from the PLLCTRL (\$08) register.

This function takes no arguments.

Returns

If successful, returns a value from 0 to 15.

Returns -1 on an error.

CMX655GetIsr

Summary

```
#include "655spt.h"

int      CMX655GetIsr(void)
```

Description

Reads back the contents of the ISR (\$00) register.

This function takes no arguments.

Returns

Returns the ISR settings.

Returns -1 on an error.

CMX655GetLevel

Summary

```
#include "655spt.h"

int      CMX655GetLevel (piLeft,piRight)

int      *piLeft;
int      *piRight;
```

Description

Reads the current left and right record Level Control from the LEVEL (\$0F) register.

piLeft points to an integer where the function may place the level for the left channel.

piRight points to an integer where the function may place the level for the right channel.

Returns

If successful returns piLeft & piRight, two 4 bit words for left and right record level, representing -12dB to +3dB (+1dB steps) for each channel.

CMX655GetLFilt

Summary

```
#include "655spt.h"

int      CMX655GetLFilt(void)
```

Description

Reads the charge Loop Filter Resistor setting from the PLLCTRL (\$08) register.

This function takes no arguments.

Returns

If successful, returns a value from 0 to 15.

Returns -1 on an error.

CMX655GetNdiv

Summary

```
#include "655spt.h"

int      CMX655GetNdiv(void)
```

Description

Reads back the contents of the feedback divider NDIVHI (\$06) and NDIVLO (\$07) registers.

This function takes no arguments.

Returns

Returns the N register value NDIV(12:0) if successful.

Returns -1 on an error.

CMX655GetNoiseGateAtten

Summary

```
#include "655spt.h"

int      CMX655GetNoiseGateAtten(piAtten)

int      *piAtten;
```

Description

Reads the current noise gate attenuation values from the NGLSTAT (\$1E) and NGRSTAT (\$1F) registers.

piAtten points to an array of at least 2 integers where the function may place the values read. piAtten[0] holds the NGLSTAT value and piAtten[1] holds the NGRSTAT value.

Returns

Returns -1 on an error.

CMX655GetPllCtrl

Summary

```
#include "655spt.h"

int      CMX655GetPllCtrl(void)
```

Description

Read the PLLCTRL (\$08) register and returns the value.

This function takes no arguments.

Returns

If successful, returns the contents of the PLL Control Register contents for Loop Filter Resistor and Charge Pump Current Gain settings, two values between 0 and 15.

Returns -1 on an error.

CMX655GetPvf

Summary

```
#include "655spt.h"

int      CMX655GetPvf(void)
```

Description

Reads the PVF (\$28) register and returns the current playback voice filter settings.

This function takes no arguments.

Returns

If successful, returns the current state of the Playback Low Pass and DC Blocking Filters as well as the High Pass Filter Selection PVF(1:0).

Returns -1 on an error.

CMX655GetRdiv

Summary

```
#include "655spt.h"

int      CMX655GetRdiv(void)
```

Description

Reads back the contents of the feedback divider RDIVHI (\$04) and RDIVLO (\$05) registers.

This function takes no arguments.

Returns

Returns the N register value RDIV(12:0) if successful.

Returns -1 on an error.

CMX655GetRvf

Summary

```
#include "655spt.h"

int      CMX655GetRvf(void)
```

Description

Reads the RVF (\$0C) register and returns the current record voice filter settings.

This function takes no arguments.

Returns

If successful, returns the current state of the Record Low Pass and DC Blocking Filters as well as the High Pass Filter Selection RVF(1:0).

Returns -1 on an error.

CMX655GetSai

Summary

```
#include "655spt.h"

int      CMX655GetSai(void)
```

Description

Reads the SAICTRL (\$09) register and returns the current serial audio interface control settings.

This function takes no arguments.

Returns

If successful, returns the current states of the MSTR(7), WL(6), MONO(5), DLY(4), POL(3), BINV(2) and PCM(0) enable bits

CMX655GetSaiM

Summary

```
#include "655spt.h"

int      CMX655GetSaiM(void)
```

Description

Reads the SAIMUX (\$0A) register and returns the current serial audio interface multiplexer settings.

This function takes no arguments.

Returns

If successful, returns the current states of the ALAW(5), COMP(4), AMP(3:2), MIC(1:0) bits.

Returns -1 on an error.

CMX655GetSampleRate

Summary

```
#include "655spt.h"

int      CMX655GetSampleRate(void)
```

Description

Gets the current sample rate from bits 6 and 5 of the CLKCTRL (\$03) register.

This function takes no arguments.

Returns

Returns 0 for 8ks/s, 1 for 16ks/s, 2 for 32ks/s and 3 for 48ks/s

Returns -1 on an error.

CMX655GetSysCtrl

Summary

```
#include "655spt.h"

int      CMX655GetSysCtrl(void)
```

Description

Reads the SYSCTRL (\$32) register and returns the current system control settings.

This function takes no arguments.

Returns

If successful, returns the current states of the ADCSR(6), SAI(5), LOUT(4), PAMP(3), AMIC(2), MICL(1) and MICR(0) control bits.

Returns -1 on an error.

CMX655GetVolume

Summary

```
#include "655spt.h"

int      CMX655GetVolume(void)
```

Description

Gets the playback volume setting from the VOLUME (\$2A) register.

This function takes no arguments.

Returns

If successful, returns a value from 0 to 127 or -1 on an error.

CMX655GetVolSmooth

Summary

```
#include "655spt.h"

int      CMX655GetVolSmooth(void)
```

Description

Gets the volume smoothing flag from bit 7 of the VOLUME (\$2A) register.

This function takes no arguments.

Returns

Returns 0 if the smoothing feature is off, returns 1 if the feature is on and returns -1 on an error.

CMX655HardReset

Summary

```
#include "655spt.h"

void CMX655HardReset(void)
```

Description

Performs Hard Reset of the CMX655D by generating a pulse low with the GPIO(24) and therefore CMX655D RSTN Hard Reset pin.

This function takes no arguments

Returns

This is a void function and therefore returns nothing.

CMX655Open

Summary

```
#include "655spt.h"

int      CMX655Open(void)
```

Description

Opens and initialises the serial port for communicating with the CMX655 device registers.

Returns

Returns -1 on error.

CMX655OpenGpio

Summary

```
#include "655spt.h"

int      CMX655OpenGPIO(void)
```

Description

Enables and sets up the necessary GPIO (GPIO24 for RSTN and GPIO25 for IRQN).

This function takes no arguments

Returns

Returns -1 on an error.

CMX655Read

Summary

```
#include "655spt.h"

int      CMX655Read(iAddr,iCount,pBytes)

int      iAddr;
int      iCount;
PBYTE    pBytes;
```

Description

Reads a number of bytes from the CMX655 device registers.

iAddr specifies the address of the register to start reading from.

iCount specifies the number of bytes to read.

pBytes specifies a pointer to the buffer where the data should be stored.

Returns

Returns -1 on an error

CMX655ReadByte

Summary

```
#include "655spt.h"

int      CMX655ReadByte(iAddr)

int      iAddr;
```

Description

Reads a single byte from a CMX655 device register.

iAddr specifies the address of the register.

Returns

If no error occurs then the value read from the register will be returned.

Returns -1 on an error.

CMX655ResetHi

Summary

```
#include "655spt.h"

void CMX655ResetHi(void)
```

Description

Sets GPIO(24) as an output and then drives it high; CMX655D Hard Reset pin also pulled high.

This function takes no arguments

Returns

This is a void function and therefore returns nothing.

CMX655ResetZ

Summary

```
#include "655spt.h"

void CMX655ResetZ(void)
```

Description

Sets GPIO(24) to an input with no pull up or down.

This function takes no arguments

Returns

This is a void function and therefore returns nothing.

CMX655SetBitsSysCtrl

Summary

```
#include "655spt.h"

int      CMX655SetBitsSysCtrl (iBits)

int      iBits;
```

Description

Sets desired bits in the SYCTRL(\$32) register without affecting any other bits. This is done by a read then write transaction.

iBits specifies the bits to be set. A 1 in any bit position causes that bit to be set in the register. Multiple bits can be set by OR-ing together the desired bits.

Returns

Returns -1 on an error.

CMX655SetClkSrc

Summary

```
#include "655spt.h"

int      CMX655SetClkSrc (iClkSrc)

int      iClkSrc;
```

Description

If successful sets the desired PLL clock source by setting appropriate enable bits in the CLKCTRL (\$03) register.

iClkSrc specifies the required option. CLKCTRL_PLL_RCLK, CLKCTRL_PLL_LRCLK, CLKCTRL_LPO. CLKCTRL_RCLK is the default option.

Returns

Returns -1 on an error.

CMX655SetCpi

Summary

```
#include "655spt.h"

int      CMX655SetCpi (iValue)

int      iValue;
```

Description

Writes the supplied charge pump current gain setting to the PLLCTRL (\$08) register.

iValue specifies the charge pump current value to be written. This should be a value between 0 ($\mu\text{A}/\text{cycle}$) and 15 ($6.2 \mu\text{A}/\text{cycle}$) inclusive.

Returns

Returns -1 on an error.

CMX655SetCsLow

Summary

```
#include "655spt.h"

void      CMX655SetCsLow(void)
```

Description

Drives the SPI CS low by changing CS to GPIO8 and driving low.

This function takes no arguments

Returns

This is a void function and therefore returns nothing.

CMX655SetCsSpi

Summary

```
#include "655spt.h"

void      CMX655SetCsSpi(void)
```

Description

Places the CS control pin back to default SPI control.

This function takes no arguments

Returns

This is a void function and therefore returns nothing.

CMX655SetLevel

Summary

```
#include "655spt.h"

int      CMX655SetLevel(iLeft,iRight)

int      iLeft;
int      iRight;
```

Description

Sets the playback volume field of the VOLUME (\$2A) register.

iLeft and iRight specifies the values to be set, two 4 bit words for left and right record level, representing -12dB to +3dB (+1dB steps) for each channel.

Returns

Returns -1 on an error.

CMX655SetLFilt

Summary

```
#include "655spt.h"

int      CMX655SetLFilt(iValue)

int      iValue;
```

Description

Writes the supplied Loop Filter Resistor setting to the PLLCTRL (\$08) register.

iValue specifies the Loop Filter Resistor value to be written. This should be a value between 0 (17.5 k Ω) and 15 (3,2 M Ω) inclusive.

Returns

Returns -1 on an error.

CMX655SetNdiv

Summary

```
#include "655spt.h"

int      CMX655SetNdiv(iValue)

int      iValue;
```

Description

Writes the supplied N divider setting to the NDIVHI (\$06) and NDIVLO (\$07) registers.

iValue specifies the NDIV(12:0) N Divider setting which is written to the two NDIV registers.

Returns

Returns -1 on an error.

CMX655SetPlayPreamp

Summary

```
#include "655spt.h"

int      CMX655SetPlayPreamp(iGain)

int      iGain;
```

Description

Writes the supplied playback preamp gain setting to the PREAMP (\$29) register.

iGain specifies the 2 bit value to be written where:

0 = PRE_0dB = +0dB
1 = PRE_6dB = +6dB
2 = PRE_12dB = +12dB
3 = PRE_18dB = +18dB

Returns

Returns -1 on an error.

CMX655SetPllCtrl

Summary

```
#include "655spt.h"

int      CMX655SetPllCtrl (iPllCtrl)

int      iPllCtrl;
```

Description

Writes the supplied Loop Filter Resistor and Charge Pump Current Gain settings to the PLLCTRL (\$08) register.

iPllCtrl specifies the loop filter resistor and charge pump current gain values each of which are 16 bit values (0 to 15).

Returns

Returns -1 on an error.

CMX655SetPvf

Summary

```
#include "655spt.h"

int      CMX655SetPvf(iFilt)

int      iFilt;
```

Description

Sets the playback voice filter settings in the PVF (\$28) register.

iFilt setting can enable the playback low pass (PVF(3) and DC blocking PVF(2)) filters as well as the high Pass Filter Selection PVF(1:0).

Returns

Returns -1 on an error.

CMX655SetRdiv

Summary

```
#include "655spt.h"

int      CMX655SetRdiv(iValue)

int      iValue;
```

Description

Writes the supplied R divider setting to the RDIVHI (\$04) and RDIVLO (\$05) registers.

iValue specifies the RDIV(12:0) N Divider setting which is written to the two RDIV registers.

Returns

Returns -1 on an error.

CMX655SetRvf

Summary

```
#include "655spt.h"

int      CMX655SetRvf(iFilt)

int      iFilt;
```

Description

Sets the playback voice filter settings in the RVF (\$0C) register.

iFilt setting can enable the playback low pass (RVF(3) and DC blocking RVF(2) filters as well as the high Pass Filter Selection RVF(1:0).

Returns

Returns -1 on an error.

CMX655SetSai

Summary

```
#include "655spt.h"

int      CMX655SetSai(iSai)

int      iSai;
```

Description

Writes to the serial audio interface control register SAICTRL (\$09).

iSai sets MSTR(7), WL(6), MONO(5), DLY(4), POL(3), BINV(2) and PCM(0) enable bits.

Returns

Returns -1 on an error.

CMX655SetSaiM

Summary

```
#include "655spt.h"

int      CMX655SetSaiM(iSaiM)

int      iSaiM;
```

Description

Writes to the serial audio interface mux register SAIMUX (\$0A).

iSaiM sets the ALAW(5), COMP(4), AMP(3:2), MIC(1:0) enable bits.

Returns

Returns -1 on an error.

CMX655SetSampleRate

Summary

```
#include "655spt.h"

int      CMX655SetSampleRate (iSampleRate)

int      iSampleRate;
```

Description

Sets the current sample rate by writing bits 6 and 5 of the CLKCTRL (\$03) register.

iSampleRate specifies the sample rate. Values of 0 for 8ks/s, 1 for 16ks/s, 2 for 32ks/s and 3 for 48ks/s may be used. Constants for these values have also been defined:-

```
SR8K
SR16K
SR32K
SR48K
```

Note: This register cannot be modified whilst the master clock is running, and if it is, no error will be returned.

Returns

Returns -1 on an error.

CMX655SetSysCtrl

Summary

```
#include "655spt.h"

int      CMX655SetSysCtrl(iSysCtrl
int      iSysCtrl;
```

Description

Writes to the system control register SYSCTRL (\$32).

iSysCtrl sets the ADCSR(6), SAI(5), LOUT(4), PAMP(3), AMIC(2), MICL(1) and MICR(0) enable control bits.

Returns

Returns -1 on an error.

CMX655SetVolume

Summary

```
#include "655spt.h"

int      CMX655SetVolume(iVolume)

int      iVolume;
```

Description

Sets the playback volume field of the VOLUME (\$2A) register.

iVolume specifies the value to be set. A value of 0 mutes the playback channel. values 91 or greater apply 0dB attenuation, and values between 1 and 91 change the attenuation in 1dB steps – 1 is -90dB, 2 is -89dB etc...

Returns

Returns -1 on an error.

CMX655SetVolSmooth

Summary

```
#include "655spt.h"

int      CMX655SetVolSmooth(iSmooth)

int      iSmooth;
```

Description

Turns the volume smoothing feature on or off by setting or clearing bit 7 of the VOLUME (\$2A) register.

iSmooth specifies whether to turn the smoothing on or off. A non-zero value will turn it on, and zero will turn it off.

Returns

Returns -1 on an error.

CMX655Write

Summary

```
#include "655spt.h"

int      CMX655Write(iAddr,iCount,pBytes)

int      iAddr;
int      iCount;
PBYTE    pBytes;
```

Description

Writes a number of bytes to the CMX655 device registers.

iAddr specifies the address of the register to start writing to.

iCount specifies the number of bytes to send.

pBytes specifies a pointer to the bytes to send.

Returns

Returns -1 on an error

CMX655WriteByte

Summary

```
#include "655spt.h"

int      CMX655WriteByte(iAddr,iData);

int      iAddr;
int      iData;
```

Description

Writes a single byte to a CMX655 device register.

iAddr specifies the address.

iData specifies the byte.

Returns

Returns -1 on an error

ConfigGpios

Summary

```
#include "655spt.h"

void ConfigGpios(void)
```

Description

Performs GPIO configuration of GPIO 24 and 25 for RSTN and IRQN use respectively.

This function takes no arguments

Returns

This is a void function and therefore returns nothing.

GetMemHandle

Summary

```
#include "655spt.h"

int      GetMemHandle(void)
```

Description

Uses "/dev/mem" to enable a access to physical memory, including peripheral GPIO

This function takes no arguments

Returns

If successful maps GPIO peripheral address in memory.

MapGpio

Summary

```
#include "655spt.h"

PGPIO    MapGpio (fdMem)

int      fdMem;
```

Description

Sets up the three peripherals registers so that they can be read from or written to.
fdMem specifies the GPIO map address.

Returns

Pointer to GPIO access in memory

CML does not assume any responsibility for the use of any algorithms, methods or circuitry described. No IPR or circuit patent licenses are implied. CML reserves the right at any time without notice to change the said algorithms, methods and circuitry and this product specification. CML has a policy of testing every product shipped using calibrated test equipment to ensure compliance with this product specification. Specific testing of all circuit parameters is not necessarily performed.

 CML Microcircuits	United Kingdom	p: +44 (0) 1621 875500	e: sales@cmlmicro.com techsupport@cmlmicro.com
	Singapore	p: +65 62888129	e: sg.sales@cmlmicro.com sg.techsupport@cmlmicro.com
	United States	p: +1 336 744 5050 800 638 5577	e: us.sales@cmlmicro.com us.techsupport@cmlmicro.com
www.cmlmicro.com			