МОДУЛИ «ОСПЧ»

БИБЛИОТЕКА ФУНКЦИЙ. МНОГОКАНАЛЬНЫЙ РЕЖИМ TDMA

Руководство программиста

Версия 1.2

Листов 36

2020-2021

СОДЕРЖАНИЕ

1. ОБЩИЕ СВЕДЕНИЯ

Комплект «libtdma SDK» предназначен для разработки ПО управления устройствами «ОСПЧ-Е3», «ОСПЧ-Е2», «ОСПЧ-Е1», «ОСПЧ-Е» и «ОСПЧ-М1» в режимах: TDMA-MCH, TDMA-SW, TDMA-DW, TDMA-ID-Evo, TDMA-ID-Inf, A-TDMA.

В комплект «libtdma SDK» включены:

– библиотека управления;

– заголовочные файлы;

– руководство программиста;

– примеры.

В заголовочных файлах представлены описания интерфейса функций, типов параметров функций, значений возврата функций. Все функции библиотеки, за небольшим исключением, возвращают код ошибки. Коды возвратов функций описаны в разделе 1. В функциях считывания значений или статуса некоторые параметры передаются по указателю. При успешном выполнении функции поля структур или параметры будут заполнены считываемыми значениями.

2. ОБЩИЕ ФУНКЦИИ

2.1. Инициализация и освобождение ресурсов

Для инициализации библиотеки необходимо вызвать функцию создания объектов работы с устройством createInstance, после чего выполнить подключение к устройству и инициализацию путем вызова функции initLibrary. Для освобождения ресурсов библиотеки предназначена функция releaseLibrary. Если предполагается управление несколькими устройствами из одного процесса, функции createInstance, initLibrary, releaseLibrary необходимо вызывать для каждого устройства.

```
int createInstance (
        UCHAR*      devNum
);
```

Параметр	**Описание**
devNum	номер устройства

```
int initLibrary (
        UCHAR*      devNum,
        char*       firmwarePath
);
```

Параметр	**Описание**
devNum	номер устройства
firmwarePath	путь к директории Firmware

```
int releaseLibrary (
        UCHAR*      devNum
);
```

Параметр	**Описание**

devNum	номер устройства

2.2. Описание последней ошибки

В случае возникновения ошибки возвращаемое значение функции библиотеки будет отличным от нуля. Определения и значения кодов ошибок содержатся в разделе 1. Для получения подробной информации об ошибке следует воспользоваться функцией getErrorDescription.

```
int getErrorDescription (
        UCHAR*      devNum,
        char*       description,
        char*       location,
        char*       dbgInfo,
        int         maxStrLen,
);
```

Параметр	**Описание**
devNum	номер устройства
description	описание последней ошибки
location	место возникновения ошибки внутри библиотеки
dbgInfo	отладочная информация для включения в лог
maxStrLen	максимальная длина строки

2.3. Установка режима TDMA

Установка режима TDMA и загрузка ПЛИС демодулятора производится функцией loadTdmaMode. Данную функцию необходимо вызвать перед вводом параметров режима.

```
int loadTdmaMode (
        unsigned char*      devNum,
        TdmaMode            mode,
```

```
        bool                force
);
```

Параметр	Описание
devNum	номер устройства
mode	режим TDMA
force	полная перезагрузка режима

2.4. Сброс демодулятора и декодера

После загрузки демодулятора и декодера выполняется сброс схем ФАПЧ. Для этого необходимо использовать функции resetDEM и resetDEC. В режимах TDMA-ID-Evo, TDMA-ID-Inf, A-TDMA требуется дополнительный сброс ФАПЧ декодера с проверкой статуса ФАПЧ функцией checkDecPLL. При успешном захвате ФАПЧ функция checkDecPLL вернет E_NO_ERROR (0).

```
int resetDem (
        UCHAR*      devNum
);
```

Параметр	Описание
devNum	номер устройства

```
int resetDec (
        UCHAR*      devNum
);
```

Параметр	Описание
devNum	номер устройства

```
int checkDecPLL (
        UCHAR       devNum
);
```

Параметр	Описание
devNum	номер устройства

2.5. Ввод лицензионного ключа

Для работы определенных режимов устройств «ОСПЧ», в частности при получении данных с выхода декодера, необходим ввод лицензионного ключа. Ввод лицензии осуществляется функцией installLicenseKey. Для устройств «ОСПЧ-Е», «ОСПЧ-М1», и режима TDMA-DW ввод ключа выполняется сразу после загрузки ПЛИС демодулятора. Для всех остальных режимов и устройств ввод ключа осуществляется после загрузки ПЛИС декодера.

```
int installLicenseKey (
        UCHAR       devNum,
        PINT64      key
);
```

Параметр	Описание
devNum	номер устройства
key	лицензионный ключ

2.6. Установка маски каналов

Включение или отключение обработки данных, а также записи сигнала в файл для каналов производится заданием маски каналов при помощи функции setChannelMask. Порядковый номер канала соответствует выставленному биту 32-разрядного числа mask.

```
int setChannelMask (
        UCHAR       DevNum,
        DWORD       Mask
);
```

Параметр	Описание
devNum	номер устройства
mask	маска каналов

2.7. Считывание служебных параметров

В некоторых случаях для идентификации устройства требуется считать его DNA-код. Для этого предназначена функция getDnaCode.

```
int getDnaCode (
        UCHAR       DevNum,
        char*       code,
        int         maxStrLen
);
```

Параметр	Описание
devNum	номер устройства
code	указатель на массив, в который при успешном выполнении будет записано значение DNA кода в строковом формате
maxStrLen	размер массива для считывания DNA кода

Узнать версию установленного мезонинного модуля можно с помощью функции getDecoderVersionOnBoard.

```
int getDecoderVersionOnBoard (
        UCHAR               devNum,
        DecoderVersion *    DecVersion
);
```

Параметр	Описание
devNum	номер устройства
decVersion	указатель на переменную, содержащую версию мезонинного

	модуля, установленного в устройство

3. ФУНКЦИИ УПРАВЛЕНИЯ ДЕМОДУЛЯТОРОМ

3.1. Настройка частот

Установка центральной частоты выполняется функцией setCentralFrequency.

```
int setCentralFrequency (
        unsigned char*      devNum,
        double              frequency
);
```

Параметр	**Описание**
devNum	номер устройства
frequency	центральная частота, Гц

Установка несущей частоты выполняется функцией setCarrierFrequency.

```
int setCarrierFrequency (
        unsigned char*      devNum,
        unsigned char       channel,
        double              frequency
);
```

Параметр	**Описание**
devNum	номер устройства
channel	номер канала обработки данных
frequency	несущая частота, Гц

Для установки тактовой частоты необходимо использовать функцию setClockFrequency.

```
int setClockFrequency (
        unsigned char*      devNum,
        unsigned char       channel,
        double              frequency
);
```

Параметр	Описание
devNum	номер устройства
channel	номер канала обработки данных
frequency	тактовая частота, кГц

Настройка несущей и тактовой частот выполняется для каждого канала демодулятора. Частоты задаются в герцах (Гц). Установка несущей частоты должна выполняться после установки центральной частоты. Несущие частоты для всех каналов не должны отличаться от тактовой частоты более чем на половину полосы L-конвертора, то есть на 36 МГц или 55МГц (в зависимости от применяемого типа L-конвертора).

3.2. Управление схемами регулировки усиления

Устройства «ОСПЧ» имеют две схемы регулировки усиления (РУ). Первая схема РУ аналоговая, настраивается для всех каналов. Вторая схема РУ цифровая, реализована в ПЛИС, настраивается для каждого канала. Схемы регулировки усиления могут работать в автоматическом (АРУ) и ручном режиме (РРУ). Для настройки первой схемы РУ необходимо использовать функцию setGC1. Управление второй схемой РУ производится функцией setGC2.

```
int setGC1 (
        UCHAR*          devNum,
        GC1Ctrl*        params
);
```

Параметр	Описание
devNum	номер устройства
params	указатель на структуру параметров первой схемы РУ

Таблица 1. Структура параметров первой схемы РУ

```
GC1Control {
        WORD        gain;
        UCHAR       manual;
        UCHAR       att;
};
```

gain	нижняя или верхняя граница "нечувствительности" АРУ
manual	включение ручного режима управления схемой РУ
att	код управления аттенюатором при РРУ

```
int setGC2 (
        UCHAR*          devNum,
        UCHAR           channel,
        GC2Ctrl*        params
);
```

Параметр	**Описание**
devNum	номер устройства
channel	номер канала обработки данных
params	указатель на структуру параметров второй схемы РУ

Таблица 2. Структура параметров второй схемы РУ

```
GC2Control {
        WORD        threshold;
        WORD        limit;
        WORD        gain;
        UCHAR       manual;
        UCHAR       timeHigh;
        UCHAR       timeLow;
};
```

Threshold	значение порога работы АРУ
Limit	ограничитель макс. усиления АРУ
Gain	код усиления при РРУ

Manual	включение ручного режима управления схемой РУ
TimeHigh	управление постоянной времени АРУ
TimeLow	управление постоянной времени АРУ

Считывание значений аттенюаторов первой и второй схемы РУ выполняется функцией getValuesGC.

```
int getValuesGC (
        UCHAR*      devNum,
        UCHAR       channel,
        GCValues*   values
);
```

Параметр	**Описание**
devNum	номер устройства
channel	номер канала обработки данных
values	указатель на структуру параметров РУ

Таблица 3. Структура параметров схем РУ

```
GCValues {
        double      ARU1Hard;
        double      ARU1Soft;
        DWORD       ARU1_YE;
        DWORD       ARU2_YE;
        double      ARU2_dB;
        bool        GC1Overload;
};
```

ARU1Hard	значение аттенюатора первой схемы РУ, дБ
ARU1Soft	значение аттенюатора первой схемы РУ, дБ
ARU1_YE	код аттенюатора первой схемы РУ
ARU2_YE	код аттенюатора второй схемы РУ
ARU2_dB	значение аттенюатора второй схемы РУ, дБ

GC1Overload	статус перегрузки схемы РУ

3.3. Загрузка фильтра

Загрузка фильтра для каждого канала демодулятора выполняется функцией loadFilter.

```
int loadFilter (
        UCHAR*      devNum,
        UCHAR       channel,
        FilterType  type
);
```

Параметр	**Описание**
devNum	номер устройства
channel	номер канала обработки данных
type	тип фильтра

3.4. Настройка ФАПЧ

Настройка ФАПЧ производится функцией setPLL.

```
int setPLL(
        UCHAR*      devNum,
        UCHAR       channel,
        PLLCtrl*    params);
```

Параметр	**Описание**
devNum	номер устройства
channel	номер канала обработки данных
params	параметры ФАПЧ

Таблица 4. Структура параметров ФАПЧ

```
PLLCtrl{
        unsigned char       pllBand;
        bool                carrierTracking;
        bool                clockTracking;
        SHORT               preambleLength;
        SHORT               slotLen;
        SIGNAL_TYPE         modulation;
};
```

pllBand	ширина полосы ФАПЧ
carrierTracking	слежение за несущей
clockTracking	слежение за тактовой
preambleLength	длина преамбулы
slotLen	длина слота
modulation	вид модуляции

3.5. Управление коррелятором

Загрузка параметров коррелятора требуется во всех режимах. Для этого предназначена функция loadCorrelator. Перед загрузкой параметров коррелятора требуется выполнить его инициализацию функцией initDemCorrelator.

```
int initDemCorrelator (
        unsigned char*      devNum
);
```

Параметр	**Описание**
devNum	номер устройства

```
int loadCorrelator (
        unsigned char*      devNum,
```

```
        unsigned char       channel,
        void*               params
);
```

Параметр	**Описание**
devNum	номер устройства
channel	номер канала обработки данных
params	параметры коррелятора

Таблица 5. Параметры коррелятора режимов TDMA-MCH, TDMA-SW, A-TDMA

```
CorrelatorCtrl {
        DWORD           uwI;
        DWORD           uwQ;
        DWORD           uwMask;
        DEC_SPEED       decSpeed;
        DWORD           packetLength;
        DWORD           treshold;
};
```

uwI	уникальное слово I
uwQ	уникальное слово Q
uwMask	маска уникального слова
decSpeed	скорость декодера
packetLength	длина пакета
treshold	порог коррелятора

Таблица 6. Параметры коррелятора режимов TDMA-DW

```
CorrelatorCtrl {
        double          dt;
};
```

dt	время запаздывания пакетов (мс). По умолчанию имеет

	значение 0.07.

Таблица 7. Параметры коррелятора режимов TDMA-ID

IDirDemCorrelatorParam {	
char uw[IDIR_MAX_UW_LEN];	
int uwLength;	
unsigned int uwI;	
unsigned int uwQ;	
bool useIQUW;	
int preambleLength;	
int burstLength;	
DWORD correlatorThreshold;	
};	
Channel	номер канала
uw	уникальное слово в строковом формате
uwLength	длина уникального слова в строковом формате
uwI	уникальное слово I
uwQ	уникальное слово Q
useIQUW	выбор типа уникального слова (0 - строковый формат, 1- формат IQ)
preambleLength	длина преамбулы
burstLength	длина пакета
correlatorThreshold	порог коррелятора

3.6. Управление счетчиком синхронизации

Для синхронизации ввода данных и добавления в сигнал временных меток в ПЛИС демодулятора предусмотрен счетчик. Счетчик стартует сразу после

загрузки ПЛИС. Управление счетчиком (остановка, перезапуск) осуществляется функцией syncCounterControl.

```
int syncCounterControl (
        unsigned char*      devNum,
        bool                enable
);
```

Параметр	**Описание**
devNum	номер устройства
enable	управление счетчиком

3.7. Считывание параметров демодулятора

При работе с «ОСПЧ» может понадобиться считать состояние демодулятора. Считать статус синхронизации можно с помощью функции readSyncStatus.

```
int readSyncStatus (
        UCHAR*              devNum,
        SyncStatus*         value
);
```

Параметр	**Описание**
devNum	номер устройства
value	указатель на структуру статуса синхронизации

Таблица 8. Структура параметров статуса синхронизации

```
PStatus_DW {
        struct  {
                bool   chSync;
                bool   uwSync [UW_COUNT_DW];
        } channel [MAX_CHANNEL_COUNT];
};
```

chSync	синхронизация канала
uwSync	синхронизация уникального слова

4. ФУНКЦИИ УПРАВЛЕНИЯ ДЕКОДЕРОМ

Для загрузки декодера необходимо загрузить ПЛИС и задать параметры декодера. Для загрузки файла конфигурации ПЛИС используется функция loadDecFPGA. Ввод параметров декодера осуществляется функцией loadDecParameters. Для режимов TDMA-ID-Evo и TDMA-ID-Inf после загрузки ПЛИС требуется выполнить загрузку ОЗУ мезонинного модуля.

4.1. Загрузка параметров декодера

Ввод параметров декодера осуществляется функцией loadDecParameters. Структура параметров декодера должна быть выбрана в соответствие с текущим режимом TDMA, заданным функцией loadTdmaMode.

```
int loadDecParameters (
        UCHAR*                  devNum,
        void*                   params
);
```

Параметр	**Описание**
DevNum	номер устройства
DecParam	указатель на структуру параметров декодера TDMA-MCH, TDMA-SW

Таблица 9. Структура параметров декодера TDMA-MCH, TDMA-SW

```
DECCtrlMchSw{
        unsigned long       allPacketLength;
        unsigned long       scramblerValue;
        bool                header;
        {
                DecSpeed            decSpeed;
                unsigned long       packetLength;
        } channel [MAX_CHANNEL_COUNT];
```

};	
allPacketLength	размер выводимых пакетов в битах
scramblerValue	значение скремблера
header	включение заголовка
decSpeed	скорость декодера
decSpeed	длина пакета

Таблица 10. Структура параметров декодера TDMA-DW

```
DECCtrlDw {
        unsigned long       allPacketLength;
        bool                scramblerEnable;
        bool                header;
};
```

allPacketLength	размер выводимых пакетов в битах
scramblerEnable	включение скремблера
header	заголовок

Таблица 11. Структура параметров декодера TDMA-ID-Evo

```
DECCtrlIdEvo {
        bool   header;
        {
                int                 payloadLengthBytes;
                int                 yPuncturePediod;
                unsigned int        yPuncturePattern;
                int                 p;
                int                 q0;
                int                 q1;
                int                 q2;
                int                 q3;
                int                 pilotPeriod;
                int                 pilotBlock;
```

```
                int                 infPilot1;
                int                 infPilot2;
                int                 infPilot3;
                int                 infPilot4;
        } channel [MAX_CHANNEL_COUNT];
};
```

header	включение заголовка
payloadLengthBytes	размер информационной части пакета
yPuncturePediod	период выкалывания проверочной части
yPuncturePattern	маска выкалывания проверочной части
p	параметр "P" перемежителя
q0	параметр "Q0" перемежителя
q1	параметр "Q1" перемежителя
q2	параметр "Q2" перемежителя
q3	параметр "Q3" перемежителя
pilotPeriod	период повторения пилот-сигналов
pilotBlock	размер блока пилот-сигналов
infPilot1	размер информационной части до 1-го пилот-сигнала
infPilot2	размер информационной части до 2-го пилот-сигнала
infPilot3	размер информационной части до 3-го пилот-сигнала
infPilot4	размер информационной части до 4-го пилот-сигнала

Параметры infPilot1, infPilot2, infPilot3, infPilot4 актуальны только для вида модуляции ФМ-8.

Таблица 12. Структура параметров декодера TDMA-ID-Inf

```
DECCtrlIdInf{
        bool   header;
        {
                INFINITY_MODE    mode;
```

```
        } channel [MAX_CHANNEL_COUNT];

};
```

header	включение заголовка
mode	режим декодера TDMA-ID-Inf

Таблица 13. Структура параметров декодера A-TDMA

```
DECCtrlATdma{
        bool   header;
        {
                ATDMA_PAYLOAD_SIZE          payloadSize;
                ATDMA_SPEED                 speed;
                ATDMA_PILOT_SIZE            pilotSize;
                unsigned char               pilotPeriod;
        } channel [MAX_CHANNEL_COUNT];
};
```

payloadSize	размер информационной части пакета
speed	кодовая скорость
pilotSize	размер блока пилот-сигналов
pilotPeriod	период повторения пилот-сигналов

Таблица 14. Структура параметров декодера SW7000

```
DECCtrlSW7000{
        bool   scramblerEn;
        bool   header;
        {
                Sw7000InfSize       infSize;
                Sw7000Speed         speed;
                unsigned char       pilotCount;
                unsigned char       codeWordCount;
                unsigned int        packetLength;
```

} channel [MAX_CHANNEL_COUNT]; };	
scramblerEn	включение скремблера
header	включение заголовка
infSize	размер информационной части пакета
speed	относительная скорость
pilotCount	количество пилот-сигналов
codeWordCount	количество кодовых слов
packetLength	длина пакета

4.2. Загрузка ОЗУ декодера

В режимах TDMA-ID-Evo, TDMA-ID-Inf, A-TDMA после загрузки ПЛИС декодера требуется выполнить загрузку ОЗУ декодера. Для этого предназначена функция loadDecRam.

```
int loadDecRam (
        UCHAR*      devNum
);
```

Параметр	**Описание**
devNum	номер устройства в системе

4.3. Установка инверсии спектра

Для всех режимов кроме TDMA-DW доступно включение инверсии спектра. Для этого предназначена функция spectrumInversion.

```
int spectrumInversion (
        UCHAR*      devNum,
```

```
        bool        value
);
```

Параметр	Описание
devNum	номер устройства
value	управление инверсией спектра

4.4. Считывание параметров декодера

В процессе работы с устройством может потребоваться информация о состоянии декодера. Считывание вероятности битовой ошибки на выходе декодера производится функцией readBER.

```
int readBER (
        UCHAR       devNum,
        UCHAR       channel,
        double*     berValue
);
```

Параметр	Описание
devNum	номер устройства
channel	номер канала
berValue	указатель на переменную, содержащую значение BER

Функция readDECTemperature возвращает температуру ПЛИС декодера, а также статус перегрева ПЛИС.

```
int readDECTemperature (
        UCHAR*      devNum,
        int*        temperature
        bool*       overheat
);
```

Параметр	**Описание**
devNum	номер устройства
temperature	указатель на переменную, содержащую температуру мезонинного модуля
overheat	указатель на переменную, содержащую статус перегрева

5. ФУНКЦИИ УПРАВЛЕНИЯ КАНАЛАМИ ДАННЫХ

Устройства «ОСПЧ» поддерживают два режима передачи данных в ОЗУ ПЭВМ: режим с захватом шины PCI и прямым доступом к памяти (Busmaster) и режим Slave. Каналы передачи данных DMA можно разделить на 2 типа: DMD и IQ.

В режимах TDMA-ID, A-TDMA, а также TDMA-MCH, TDMA-SW, TDMA-DW для «ОСПЧ-Е3» с типом L-конвертора, выше или равным 5, каждый канал обработки данных демодулятора имеет свой канал DMA передачи демодулированных и декодированных данных (DMD). При этом номер канала DMA соответствует номеру канала обработки сигнала демодулятора, умноженному на 2. В остальных случаях, данные всех каналов обработки данных передаются по нулевому каналу DMA.

Канал IQ общий для всех каналов обработки данных демодулятора. Данный DMA канал (IQ) предназначен для получения данных АЦП и отсчетов для построения векторной диаграммы выбранного канала обработки данных демодулятора. Для данных АЦП или IQ номер DMA канала равен 1.

Настройка каналов данных выполняется функцией setDataChannel.

```
int setDataChannel (
        UCHAR*              devNum,
        UCHAR               channel,
        DataChannelCtrl*    params
);
```

Параметр	**Описание**
devNum	номер устройства
channel	номер канала обработки данных

params	указатель на структуру параметров канала данных

Таблица 15. Структура параметров канала данных

```
DataChannelCtrl {
        DataFormat      format;
        bool            transferEn;
        bool            singleMode;
        bool            reverseBits;
        bool            spectrumMode;
};
```

format	управление форматом сигнала при записи
transferEn	пересылка данных в режиме Busmaster
singleMode	режим однократной пересылки
reverseBits	разворот бит в байте
spectrumMode	передача данных для построения спектра

6. ВОЗВРАЩАЕМЫЕ ЗНАЧЕНИЯ

Идентификатор		**Описание**
E_NO_ERROR	0	Выполнено без ошибок
E_INSTANCE_NOT_INIT	-101	Объекты, необходимые для работы библиотеки не созданы или не проведена инициализация
E_LICENSE_NOT_ACCEPTED	-111	Лицензионный ключ не принят
E_MODE_NOT_SUPPORTED	-112	На данной аппаратной конфигурации режим не поддерживается или еще не реализован
E_INCORRECT_PARAMETERS	-113	Неправильные параметры, или параметры вне диапазона допустимых значений
E_FPGA_DEC_NOT_LOADED	-114	Плис декодера не загружена, сначала нужно выполнить loadDecFpga()
E_OPEN_FILE_ERROR	-117	Не удалось открыть файл
E_INTERNAL_ERROR	-120	Внутренняя ошибка
E_UNHANDLED_EXCEPTION	-121	Исключение при выполнении кода библиотеки
E_DEC_PLL_NOT_LOCKED	-122	Нет захвата PLL декодера (PLL не сбросилась)
E_DEVICE_ALREADY_CONNECTED	-131	Устройство уже подключено, попытка подключения к занятому устройству
E_DEVICES_NOT_FOUND	-132	Не найдено поддерживаемых устройств ОСПЧ
E_DEVICE_TYPE_NOT_SUPPORTED	-133	Устройство данного типа не поддерживается
E_NO_OSPCHLIB_DEFINED	-141	Для данного типа устройства и среды запуска библиотека ОСПЧ не определена
E_OSPCHLIB_BAD_INTERFACE	-142	Некоторые функции библиотеки ОСПЧ не были загружены
E_OSPCHLIB_ERROR	-143	Ошибка библиотеки ospchx.dll
E_OSPCHLIB_EXCEPTION	-144	Исключение при вызове функции библиотеки ospchx.dll
E_IO_ERROR	-151	Ошибка ввода-вывода (при выполнении _devoutp/_devinp)

E_IO_EXCEPTION	-152	Исключение ввода-вывода (при выполнении _devoutp/_devinp)
E_FPGA_LOAD_ERROR	-161	Ошибка загрузки ПЛИС (fpgaLoad вернула не 0)
E_DECODER_NOT_SUPPORTED	-163	Тип мезонинного модуля не поддерживается или неправильно определен
E_SN_READ_ERROR	-165	Ошибка считывания серийного номера
E_DNA_READ_ERROR	-166	Ошибка считывания кода DNA
E_DNA_NOT_INITED	-167	DNA код не инициализирован
E_DNA_NOT_SUPPORTED	-168	DNA код не поддерживается (поддержка только для ОСПЧ-Е3)
E_UNHANDLED_DEVICE_ERROR	-180	Прочие ошибки устройства

7. РЕКОМЕНДУЕМЫЙ ПОРЯДОК РАБОТЫ

7.1. Подключение к устройству

7.1.1. Определить число поддерживаемых устройств: *DeviceSupportedCount*

7.1.2. Найти устройство нужного типа, используя для получения имени устройства в системе функцию *GetFullDeviceName*

7.1.3. Инициализировать выбранное устройство: *CreateDeviceInstance*

7.1.4. Для устройств «ОСПЧ-Е1», «ОСПЧ-Е2», «ОСПЧ-Е3» выполнить загрузку синтезатора: *rSynthLoad*

7.1.5. Инициализировать библиотеку управления: *createInstance*, *initLibrary*

7.1.6. Получить для дальнейшего использования базовые параметры устройства: *GetDeviceSlotId*, *GetLConvertorType*, *GetLConvertorVersion*, *GetDecoderVersionOnBoard*

7.1.7. Считать из реестра количество буферов памяти на канал и максимальные значения размеров буферов

7.2. Загрузка демодулятора

7.2.1. Остановить DMA мастер: *MasterStop*

7.2.2. Загрузить режим TDMA: *loadTdmaMode*

7.2.3. При необходимости установить лицензионный ключ: *InstallLicenseKey*

7.2.4. Загрузить АЦП: *ADCLoadMulti*

7.2.5. Для устройств «ОСПЧ-Е1», «ОСПЧ-Е2», «ОСПЧ-Е3» выполнить загрузку синтезатора L-конвертора и чтение таблицы управляющих напряжений: *lSynthInit*, *InitLConvertorVList*

7.2.6. Установить центральную частоту: *setCentralFrequency*

7.2.7. Настроить первую схему РУ: *setGC1*

7.2.8. Инициализировать коррелятор: *initDemCorrelator*

7.2.9. Для каждого канала демодулятора:

7.2.9.1. Настроить вторую схему РУ: *setGC2*

7.2.9.2. Установить несущую частоту: *setCarrierFrequency*

7.2.10. Для каждого канала демодулятора:

7.2.10.1. Установить тактовую частоту: *setClockFrequency*

7.2.10.2. Загрузить фильтр: *loadFilter*

7.2.11. Настроить PLL для каждого канала демодулятора: *setPLL*

7.2.12. Загрузить коррелятор для каждого канала демодулятора: *loadCorrelator*

7.3. Загрузка декодера на примере TDMA-ID-Inf

7.3.1. Загрузить ПЛИС декодера: *loadDecFPGA*

7.3.2. Ввести лицензионный ключ: *InstallLicenseKey*

7.3.3. Сбросить ФАПЧ декодера: *resetDec*

7.3.4. Проверить захват ФАПЧ: *checkDecPll*

7.3.5. Загрузить ОЗУ декодера: *loadDecRam*

7.3.6. Ввести параметры декодера режима IDMA-ID-Inf: *loadDecParameters*

7.3.7. Настроить инверсию спектра: *spectrumInversion*

7.3.8. Сбросить ФАПЧ декодера: *resetDEC*

7.3.9. Сбросить ФАПЧ демодулятора: *resetDEM*

7.3.10. Установить маску каналов: *setChannelMask*

7.4. Настройка буферов памяти для получения данных

7.4.1. Остановить Busmaster: *MasterStop*

7.4.2. Получить адреса буферов: *MasterGetBuffers*

7.4.3. Установить размер буферов

7.5. Запись сигнала в непрерывном режиме

7.5.1. Остановить Busmaster: *MasterStop*

Примечание: функция MasterLoad должна вызываться при остановленной пересылке данных в режиме Busmaster.

7.5.2. Выполнить загрузку Busmaster: *MasterLoad*

7.5.3. Запустить Busmaster: *MasterStart*

7.5.4. Инициализировать чтение данных

Подготовить поток (thread) получения данных. В потоке использовать событие с именем "Global\\dgspm[deviceslot]evnt[channel]", где [deviceslot] – слот, полученный ранее, [channel] – номер канала DMA.

7.5.5. Включить передачу данных: *setDataChannel*

7.5.6. Считывать данные в ранее инициализированном потоке

7.5.7. По окончании ввода данных необходимо выключить передачу данных и остановить Busmaster: *setDataChannel*, *MasterStop*

Примечание: перед вводом данных из устройства необходимо убедиться, что драйвер настроен на необходимое количество каналов. Если это не так (как правило, по умолчанию для одноканальной версии используется 4

канала), необходимо выполнить настройку каналов в реестре и перезагрузить компьютер.

7.6. Завершение работы

7.6.1. Деинициализировать библиотеку управления: *releaseLibrary*

7.6.2. Отключиться от устройства: *ReleaseDeviceInstance*

8. ПЕРЕЧЕНЬ СОКРАЩЕНИЙ

АПЧ	автоматическая подстройка частоты
АРУ	автоматическая регулировка усиления
ОС	операционная система
ОЗУ	оперативное запоминающее устройство
ОСШ	отношение сигнал-шум
ПО	программное обеспечение
ПЭВМ	персональная электронно-вычислительная машина
РРУ	ручная регулировка усиления
РУ	регулировка усиления
ФАПЧ	фазовая автоподстройка частоты