# SDK API User's Guide (Subtitle Parser)

Version 0.1.0

## Display Audio

Solution Team

## Release information

The following changes have been made to this document.

**Change History**

| Date | Change |
|---|---|
| 06 Dec. 2017 | First release for v0.1.0 |

## Proprietary Notice

Information in this document is provided solely to enable system and software implementers to use Nexell products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document.

Nexell reserves the right to make changes without further notice to any products herein.

Nexell makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Nexell assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. “Typical” parameters which may be provided in Nexell data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including “Typicals” must be validated for each customer application by customer’s technical experts. Nexell does not convey any license under its patent rights nor the rights of others. Nexell products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Nexell product could create a situation where personal injury or death may occur. Should Buyer purchase or use Nexell products for any such unintended or unauthorized application, Buyer shall indemnify and hold Nexell and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Nexell was negligent regarding the design or manufacture of the part.

**Contact us**

[11595] BundangYemiji Bldg. 12F, 31 Hwangsaeul-ro 258 beon gil, Bundang-gu, Sungnam-city, Gyeonggi-do, Korea.

TEL: 82-31-698-7400

FAX:82-31-698-7455

http://www.nexell.co.kr

# Contents

# Chap 1. Overview

## 1.1 Overview

This document describes how to use subtitle parser API.

Library name : libnxsubtitleparser.so

Class name : CNX_SubtitleParser

Header : CNX_SubtitleParser.h

### 1.1.1 List of CNX_SubtitleParser

```
struct PARSED_SUBTITLE{
         int startTime;
         int endTime;
         char* subtitleTextString;
}

struct CODECLIST{
         const char* encode;
         int confidence;
}

CNX_SubtitleParser()
virtual ~CNX_SubtitleParser()

int NX_SPOpen()

void NX_SPIncreaseIndex()
int NX_SPGetMinIndex()
int NX_SPGetMaxIndex()
int NX_SPGetIndex()
void NX_SPSetIndex(int idx)

int NX_SPGetStartTime()
int NX_SPGetEndTime()
char* NX_SPGetSubtitle()

int NX_SPGetSubtitleSync()
```

```
void NX_SPChangeSubtitleSync(int milliseconds)
int NX_SPSeekSubtitleIndex(int milliseconds)

PARSED_SUBTITLE NX_SPGetParsedSubtitleArray()
PARSED_SUBTITLE NX_SPGetParsedSubtitleArray(int index)

int NX_SPGetCodecList( CODECLIST ** codec )
const char* NX_SPGetBestEncode()

int NX_SPClose()
bool NX_SPIsParsed()
```

## 1.2 Supporting range

### 1.2.1 SAMI(smi)

Syncrhonized Accessible Media Interchange

**Structure**

```
<SAMI>
<HEAD>
<TITLE> ... </TITLE>
<STYLE TYPE="text/css"> ... </STYLE>
</HEAD>
<BODY>
...
</BODY>
</SAMI>
```

**Example**

```
<SAMI>
<HEAD>
<TITLE> mediaTitle </TITLE>
<STYLE TYPE=”text/css”>
<!–
P { font-family: Arial; font-weight: normal; color: white; background-color: black;
Text-align: center; }
 .ENUSCC { name: English; lang: en-US ; SAMIType: CC ; }
-->
</STYLE>
</HEAD>
<BODY>
<SYNC Start=2220>
```

```
<P Class=ENUSCC>How are you<br>I’m good, and you?</P>
</SYNC>
<SYNC Start=4220>
<P Class=ENUSCC>So, what’s up?</P>
</SYNC>
...
</BODY>
</SAMI>
```

SAMI format is similar to HTML and CSS.

### 1.2.2 SubRip Title(srt)

```
Structure
Subtitle number
Start time --> end time
Subtitle...
blank line

…
```

```
Example
1
00:02:22,440 --> 00:02:25,375
hi

2
00:02:26,440 --> 00:02:27,375
Hello


3
…
```

## 1.3 Environment

### 1.3.1 CPU

S5P4418

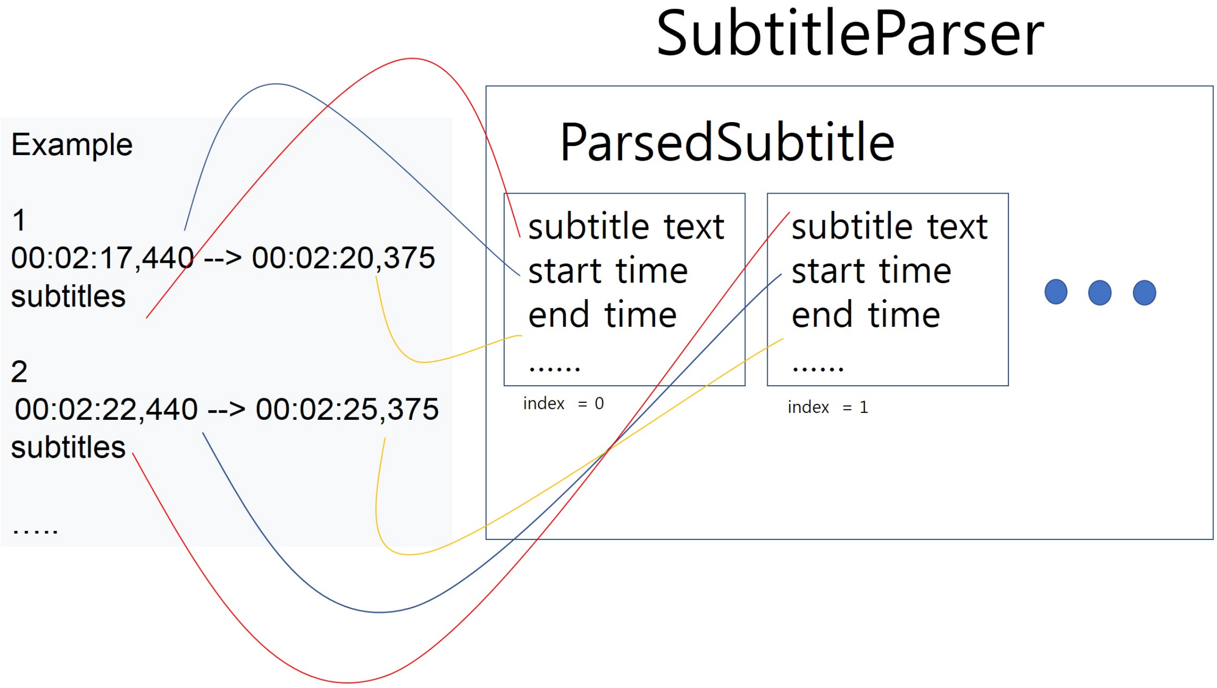
### 1.3.2 OS

Linux

# Chap 2. **Structure**

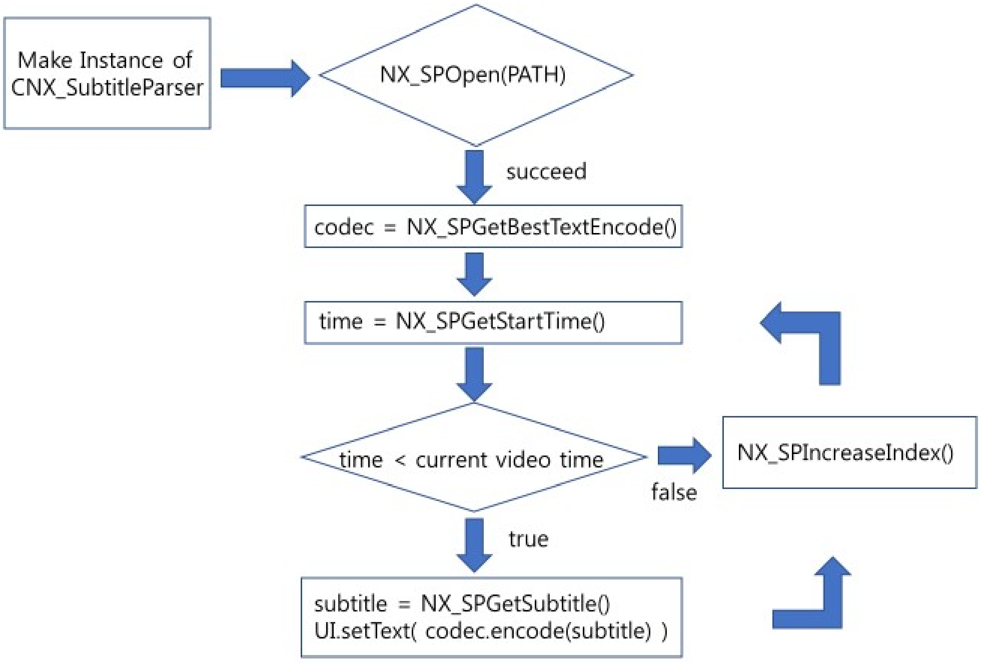
## 2.1 Structure

When subtitle parsing is done, subtitle information is stored as below.



# Chap 3. Scenario

## 3.1 Single Language


Make Instance of CNX_SubtitleParser
NX_SPOpen(PATH)
succeed
codec = NX_SPGetBestTextEncode()
time = NX_SPGetStartTime()
time < current video time
false
NX_SPIncreaseIndex()
true
subtitle = NX_SPGetSubtitle()
UI.setText( codec.encode(subtitle) )

# Chap 4. APIS

## 4.1 Overview

A detailed description of the API.

## 4.2 API Details

### 4.2.1 NX_SPOpen

**Prototype**

```
int      NX_SPOpen(const char* fullpath)
```

**Description**

This function opens subtitle file and parses subtitle information.

**Arguments**

Path of subtitle file as const char*.

**Return Value**

```
  1 is returned, if opening file and parsing subtitle are successfully done.
 -1 is returned, if fopen is failed(check path).
 -2 is returned, if memory allocation of buffer is failed(check file size).
 -3 is returned, if file is neither srt nor smi.
 -4 is returned, if file is detected as smi but, no available contents
                  (check syntax of file).
 -5 is returned, if file is detected as srt but, no available contents
                  (check syntax of file).
```

### 4.2.2 NX_SPIncreaseIndex

**Prototype**

```
void     NX_SPIncreaseIndex()
```

**Description**

This function increases inteager index by +1.

**Arguments**

void

**Return Value**

void

### 4.2.3 NX_SPGetMaxIndex

**Prototype**

`int       NX_SPGetMaxIndex()`

**Description**

This function returns inteager maximum index.

**Arguments**

Void

**Return Value**

If there exists subtitle text parsed, returns maximum index of parsed subtitle text.

Otherwise, returns 0.

### 4.2.4 NX_SPGetIndex

**Prototype**

`int       NX_SPGetIndex()`

**Description**

This function returns inteager current index.

**Arguments**

void

**Return Value**

If there exists subtitle text parsed, returns current index of parsed subtitle text.

Otherwise, returns 0.

### 4.2.5 NX_SPSetIndex

**Prototype**

`void      NX_SPSetIndex(int idx)`

**Description**

This function sets current index as input inteager idx.

**Arguments**

inteager value for index.

**Return Value**

void

### 4.2.6 NX_SPGetStartTime

**Prototype**

`int       NX_SPGetStartTime()`

**Description**

This function returns start time of parsed subtitle text corresponding current index.

**Arguments**

void

**Return Value**

If there exists subtitle text parsed, returns start time of parsed subtitle text corresponding current index.

Otherwise, returns 0.

### 4.2.7 NX_SPGetEndTime

**Prototype**

```
int     NX_SPGetEndTime()
```

**Description**

This function returns end time of parsed subtitle text corresponding current index.

**Arguments**

void

**Return Value**

If there exists subtitle text parsed from srt, returns end time of parsed subtitle text corresponding current index.

If there exists subtitle text parsed from smi, returns (start time-1) of next subtitle in milliseconds.

Otherwise, returns 0.

### 4.2.8 NX_SPGetSubtitle

**Prototype**

```
char* NX_SPGetSubtitle()
```

**Description**

This function returns parsed subtitle text corresponding current index.

**Arguments**

Void

**Return Value**

If there exists subtitle text parsed, returns it corresponding current index.

Otherwise, returns NULL.

### 4.2.9 NX_SPGetSubtitleSync

**Prototype**

```
int     NX_SPGetSubtitleSync()
```

**Description**

This function returns sync time.

**Arguments**

void

**Return Value**

If sync time is set by NX_SPChangeSubtitleSync function, returns it.

Otherwise, returns 0.

### 4.2.10 NX_SPChangeSubtitleSync

**Prototype**

**void NX_SPChangeSubtitleSync(int milliseconds)**

**Description**

This function sets sync time for all parsed result.

**Arguments**

inteager milliseconds

**Return Value**

void

### 4.2.11 NX_SPSeekSubtitleIndex

Prototype

**Int NX_SPSeekSubtitleIndex(int milliseconds)**

Description

Find index of parsed result by input milliseconcds.

Arguments

Inteager milliseconds that position to seek.

Return Value

If there exists parsed result to seek, this funciton returns corresponding inteager index.

Otherwise, this function returns 0.

### 4.2.12 NX_SPGetParsedSubtitleArray

**Prototype**

**PARSED_SUBTITLE NX_SPGetParsedSubtitleArray()**

**Description**

The purpose of this function is obtaining structure of parsed result corresponding m_iCurrentIndex.

**Arguments**

void

**Return Value**

```
Struct PARSED_SUBTITLE{
        int startTime;
        int endTime;
        char* subtitleTextString;
}
```

### 4.2.13 NX_SPGetParsedSubtitleArray

**Prototype**

```
PARSED_SUBTITLE   NX_SPGetParsedSubtitleArray(int index)
```

**Description**

The purpose of this function is obtaining structure of parsed result by index.

**Arguments**

Inteager index of parsed result.

**Return Value**

```
Struct PARSED_SUBTITLE{
        int startTime;
        int endTime;
        char* subtitleTextString;
}
```

### 4.2.14 NX_SPGetTEXTCODECLIST

**Prototype**

```
int       NX_SPGetTEXTCODECLIST( TEXTCODECLIST ** codec )
```

**Description**

The purpose of this function is obtaining all text encode founded by ICU library

**Arguments**

```
Struct TEXTCODECLIST{
        const char* encode;
        int confidence;
}
```

**Return Value**

If ICU library is worked successfully, return value is the number of possible text encode.

If ICU library is not worked, returns -1

### 4.2.15 NX_SPGetBestTextEncode

**Prototype**

```
const char*      NX_SPGetBestTextEncode()
```

**Description**

This function returns the best text encode founded by ICU library.

**Arguments**

void

**Return Value**

If ICU library is worked successfully, return value is the best text encode.

If ICU library is not worked, returns “EUC-KR”.

### 4.2.16 NX_SPClose

**Prototype**

```
void     NX_SPClose ()
```

**Description**

This function frees variables.

**Arguments**

void

**Return Value**

void

### 4.2.17 NX_SPIsParsed

**Prototype**

```
bool     NX_SPIsParsed()
```

**Description**

This function tells if NX_SPOpen function is successfully done.

**Arguments**

void

**Return Value**

true if subtitle is parsed.

# Chap 5. Known Issues

## 5.1 To Do List

Multi-Language function