

# Binary decoder

## Obsah

1. Popis.....	1
2. Závislosti .....	1
3. Implementačné informácie .....	1
4. Piny .....	2
5. Konfigurácia .....	2
5.1 Konfiguračná štruktúra.....	2
5.2 Konfiguračný dialóg.....	2
6. Definícia triedy .....	2
7. Príklad použitia .....	3

## 1. Popis



Trieda *Binary decoder* dekoduje dáta zakódované triedou *Binary encoder*. Vstupom sú dáta zakódované v zhustenej binárnej podobe. Výstupom je pole celých čísel.

## 2. Závislosti

Trieda *Binary decoder* používa tieto externé definičné súbory a knižnice:

**Headers:** *filtergraph.h*, *datatypes.h*, *bitstream.h*

**Libs:** *filtergraph.lib*, *bitstream.lib*

## 3. Implementačné informácie

Informácie o triede:

**Názov:** *Binary decoder*

**Verzia:** *1.0*

**Magic:** *180*

Informácie o definíciách:

**Header:** *binarydecoder.h*

**Konfiguračný header:** *binarydecodercfg.h*

**Lib:** *binarydecoder.lib*

Informácie o knižnici obsahujúcej triedu:

**Názov knižnice:** *Binary decoder Library*

**Verzia knižnice:** *1.0*

**Dll súbor knižnice:** *binarydecoder.dll*

## 4. Piny

Piny sú popísané spôsobom: „*[index pinu na filtri] názov pinu: popis pinu*“.

**[0] Cfg in:** Konfiguračný vstup. Vstupom je serializovaná konfigurácia.

**[1] Cfg out:** Konfiguračný výstup. Výstupom je serializovaná konfigurácia.

**[2] In:** Dátový vstup. Vstupom je buffer binárnych dát zakódovaný triedou *Binary encoder*.

**[3] Out:** Dátový výstup. Výstupom je pole celých čísel dekódované z binárnych dát, zapísané v štruktúre *TIntArray*.

## 5. Konfigurácia

### 5.1 Konfiguračná štruktúra

Konfiguračná štruktúra je definovaná nasledovne:

```
typedef struct {  
} TBinaryDecoderConfig;
```

Táto trieda nemá žiadne nastavenia. Výstup deterministicky závisí od zakódovaných dát vo vstupe.

### 5.2 Konfiguračný dialóg



Konfiguračný dialóg neposkytuje žiadne voľby.

## 6. Definícia triedy

Trieda *Binary decoder* je definovaná nasledovne:

```
class TBinaryDecoderFilter : public TFilter {  
private:  
    HINSTANCE hInstance;           // identifikácia inštancie knižnice  
  
    TBuffer    Input;              // vstupné binárne dáta  
    TIntArray  Output;            // výstupné pole celých čísel  
  
    TPin* CfgInPin;                // vstupný konfiguračný pin  
    TPin* CfgOutPin;              // výstupný konfiguračný pin  
    TPin* InPin;                  // vstupný pin
```

```

TPin* OutPin;                // výstupný pin

bool StopFlag;               // indikátor zastavenia

int decode();                 // dekódovacia funkcia

int setConfigFromPin(TPin *pin);    // nastavenie konfigurácie z pinu
int putConfigOnPin(TPin *pin);     // sprístupnenie konfigurácie na pine
int freeConfigOnPin(TPin *pin);    // uvoľnenie konfigurácie na pine
int clearOutputPins();            // vyčistenie výstupných pinov
int freeOutputData();             // uvoľnenie výstupných dát

public:
    TBinaryDecoderFilter();        // konštruktor
    TBinaryDecoderFilter(HINSTANCE hInst); // konštruktor s identifikáciou knižnice
    ~TBinaryDecoderFilter();       // deštruktor

int setConfigData(TBuffer config,int type);    // nastavenie konfigurácie
TBuffer getConfigData(int type);              // vrátenie konfigurácie
int initialize();                             // inicializácia
int run();                                    // beh
int reset();                                 // reset stavu
int stop();                                 // zastavenie behu
int finalize();                              // finalizácia
int showConfigDialog();                     // vyvolanie konfiguračného dialógu
};

```

## 7. Príklad použitia

