NSIGII Polygatic Video Codec — Usage Guide

Format: `.nsigii`
Version: 7.0.0
Protocol: Human Rights Verification System
Author: OBINexus Computing
Date: 17 February 2026

What is a `.nsigii` file?

A `.nsigii` file is a **constitutionally-encoded video container** produced by the NSIGII Polygatic Video Codec. It is not a standard video file and cannot be opened directly by VLC, Windows Media Player, or any conventional video player.

Each `.nsigii` file encodes video through a **trident channel architecture**:

```
TRANSMIT (Ch.0) → RECEIVE (Ch.1) → VERIFY (Ch.2)
```

Every frame carries cryptographic integrity metadata, RWX permission flags, bipartite consensus verification scores, and a human rights tag (`NSIGII_HR_TRANSMIT`). The format is an **LTF (Linkable Then Format/File)** artifact — its structure is constitutionally bound at link time, not at binary load time.

Quick Start

Encode a video

```powershell
# LTF pipe mode (recommended)
'.\your_video.mp4' | go run .\main.go

# Explicit flag mode
go run .\main.go -input your_video.mp4

# With custom dimensions
go run .\main.go -input your_video.mp4 -width 1920 -height 1080

# Custom output name
go run .\main.go -input your_video.mp4 -output my_output.nsigii
```

Output is written to `<input_name>.nsigii` in the current directory by default.

Prerequisites

Tool	Version	Install
Go	1.21+	https://go.dev/dl
FFmpeg	8.x	winget install --id Gyan.FFmpeg -e
FFprobe	(bundled with FFmpeg)	included above

After installing FFmpeg, refresh your PATH:

```powershell
$env:PATH = [System.Environment]::GetEnvironmentVariable("PATH","Machine") + ";" + [System.Environment]::GetEnvir
```

File Format Specification

Container Header (32 bytes)

```
Offset  Size  Field        Description
------  ----  -----        -----------
0       8     Magic        "NSIGII\0\0" — format identifier
8       8     Version      "7.0.0\0\0\0" — codec version
16      4     Width        uint32, frame width in pixels
20      4     Height       uint32, frame height in pixels
24      4     FrameCount   uint32, total number of encoded frames
28      4     Reserved     uint32, reserved for future use
```

Frame Structure (per frame)

```
Offset  Size  Field        Description
------  ----  -----        -----------
0       4     FrameSize    uint32, byte length of compressed frame data
4       N     FrameData    DEFLATE-compressed YUV420 frame
```

Verification Metadata (embedded per frame)

Each frame passes through three trident channels. The following state is tracked per frame:

Field	Values	Meaning
`RWXFlags`	`0x02` / `0x04` / `0x07`	WRITE → READ → EXECUTE permission chain
`DiscriminantState`	ORDER / CONSENSUS / CHAOS	$\Delta = B^2 - 4AC$ result
`WheelPosition`	0° / 120° / 240° / 360°	Trident channel progression
`HumanRightsTag`	`"NSIGII_HR_TRANSMIT"`	Preserved through full pipeline
`SequenceToken`	uint32	Bipartite even/odd order token

Reading a `.nsigii` File

Inspect the header (Go)

```go
```

```
package main

import (
    "encoding/binary"
    "fmt"
    "os"
)

func main() {
    f, _ := os.Open("output.nsigii")
    defer f.Close()

    var magic [8]byte
    var version [8]byte
    var width, height, frameCount, reserved uint32

    binary.Read(f, binary.LittleEndian, &magic)
    binary.Read(f, binary.LittleEndian, &version)
    binary.Read(f, binary.LittleEndian, &width)
    binary.Read(f, binary.LittleEndian, &height)
    binary.Read(f, binary.LittleEndian, &frameCount)
    binary.Read(f, binary.LittleEndian, &reserved)

    fmt.Printf("Magic:      %s\n", magic[:6])
    fmt.Printf("Version:    %s\n", version[:5])
    fmt.Printf("Dimensions:  %dx%d\n", width, height)
    fmt.Printf("Frames:     %d\n", frameCount)
}
```

Inspect the header (Python)

```
python
```

```
import struct

with open("output.nsigii", "rb") as f:
    magic    = f.read(8)
    version  = f.read(8)
    width    = struct.unpack("<I", f.read(4))[0]
    height   = struct.unpack("<I", f.read(4))[0]
    frames   = struct.unpack("<I", f.read(4))[0]
    reserved = struct.unpack("<I", f.read(4))[0]

print(f"Magic:      {magic[:6].decode()}")
print(f"Version:    {version[:5].decode()}")
print(f"Dimensions: {width}x{height}")
print(f"Frames:     {frames}")
```

Inspect the header (PowerShell)

```powershell
$bytes = [System.IO.File]::ReadAllBytes(".\NSIGII Living without Breathing is Suffering.nsigii")
$magic   = [System.Text.Encoding]::ASCII.GetString($bytes[0..5])
$version = [System.Text.Encoding]::ASCII.GetString($bytes[8..12])
$width   = [BitConverter]::ToUInt32($bytes, 16)
$height  = [BitConverter]::ToUInt32($bytes, 20)
$frames  = [BitConverter]::ToUInt32($bytes, 24)

Write-Host "Magic:      $magic"
Write-Host "Version:    $version"
Write-Host "Dimensions: ${width}x${height}"
Write-Host "Frames:     $frames"
```

Decoding / Playback

A decoder is not yet included in this release. To play back `.nsigii` content:

Option 1 — Decode to raw RGB24 then pipe to FFplay

Write a decoder that reverses the trident pipeline:

```
.nsigii → DEFLATE decompress → YUV420 → RGB24 → stdout
```

Then pipe to FFplay:

```powershell
```

```
.\nsigii-decoder.exe --input output.nsigii | `
  ffplay -f rawvideo -pix_fmt rgb24 -video_size 1280x720 -
```

Option 2 — Decode to MP4

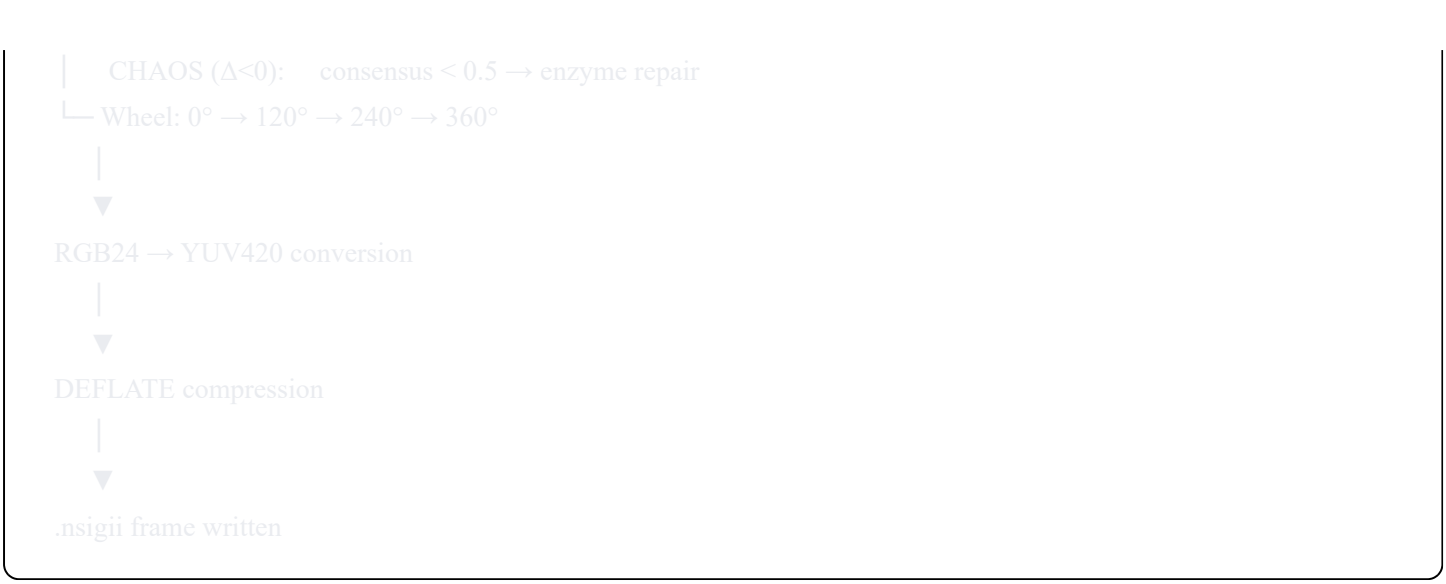
```
powershell

.\nsigii-decoder.exe --input output.nsigii | `
  ffmpeg -f rawvideo -pix_fmt rgb24 -video_size 1280x720 -r 30 -i - `
         -c:v libx264 -pix_fmt yuv420p decoded_output.mp4
```

Option 3 — Extract raw YUV420 frames

Each frame in the `.nsigii` container is DEFLATE-compressed YUV420. Read each `FrameSize` uint32, then `inflate` the following `FrameSize` bytes to recover the raw YUV420 plane data.

Encoding Pipeline Reference



Verification States

State	Discriminant	Consensus	Meaning
ORDER	Δ > 0	> 0.5	System coherent, full permissions granted
CONSENSUS	Δ = 0	= 0.5	Flash point — perfect verification
CHAOS	Δ < 0	< 0.5	Enzyme repair applied, frame still encoded

A frame in CHAOS state is not dropped — the bipolar enzyme model applies XOR-shift repair and continues encoding. Only fully corrupted frames (size mismatch) are skipped.

Compression Characteristics

Based on the reference encode (NSIGII Living without Breathing is Suffering.mp4, 1280×720, 1619 frames):

Stage	Size	Reduction
Raw RGB24	4,476,211,200 bytes	baseline
YUV420	~2,238,105,600 bytes	50%
ROPEN duplex	~1,119,052,800 bytes	75%
DEFLATE	563,376,792 bytes	**87.41%**

Integration with OBINexus Toolchain

The `.nsigii` format is part of the OBINexus LTF pipeline:

```
riftlang.exe → .so.a → rift.exe → gosilang → nsigii-codec
                                    (produces .nsigii)

Orchestration: nlink → polybuild
```

The codec acts as the **constitutional execution boundary** — the `.nsigii` output is the verified artifact, not the raw video. Each encoded file carries its verification receipt (RWX permission chain + human rights tag) as part of the container's embedded trident state.

Troubleshooting

`Failed to open input file: open video.rgb24` You are running an older version of `main.go`. Replace with the latest version which supports LTF pipe mode and the `-input` flag.

`ffprobe: executable file not found in %PATH%` Install FFmpeg: `winget install --id Gyan.FFmpeg -e --source winget`, then refresh PATH (see Prerequisites above).

`Broken pipe` errors from FFmpeg These are expected in older versions when the Go reader closes before FFmpeg finishes. The latest `main.go` drains the pipe cleanly before calling `ffmpegCmd.Wait()`.

All frames showing `CHAOS:0` during encoding is normal — CHAOS count of 0 means 100% of frames achieved ORDER or CONSENSUS state via bipartite consensus verification.

Output file has 0 frames The input video dimensions do not match the codec's frame size expectation. Run with no `-width`/`-height` flags to let ffprobe auto-detect.

File Naming Convention

Output files are named after the input file:

```
Input:   NSIGII Living without Breathing is Suffering.mp4
Output:  NSIGII Living without Breathing is Suffering.nsigii
```

The `.nsigii` extension signals that the file has passed constitutional trident verification and is an OBINexus-encoded artifact.

References

- ROPEN Specification: https://github.com/obinexus/ropen

- NSIGII BiPolar Sequence Theory (30 Jan 2026)
- Rectorial Reasoning Rational Wheel Framework (11 Feb 2026)
- Trident Command & Control Architecture
- OBINexus Constitutional Computing Framework

"Structure is a signal. Polarity is a strategy. NSIGII is the experiment."
— OBINexus Computing, 2026