Thermal Debug Requirements

This page was intentionally left blank

Debug Target Requirements

Class Name: ThermalDbgStream

Overview: The goal of the Debug Target is to provide a code library / class that allows you to integrate advanced algorithmic debugging directly into your project. Where the goal is to intercept bit-wise and mathematical operations and write into a formatted file the changes in data relative to an output buffer.

Requirements:

1. Needs to be able to create an “algorithm stream”
2. Use terminal colors to track permutations and \r to animate them.
3. Need to be able to create “steps” and add them to an algorithm stream
4. Need to be able to add “operation” to a step
5. need to be able to mark the end of adding operations to a step, generate ted data.
6. Need to be able to mark the end of a step, generate ted data.
7. Need to be able to mark the end of a stream, generate ted data and write to file.
8. Easy dump disable

Further Breakdown:

Algorithm Stream -

An algorithm stream is made of:

1. Algorithm Name
2. Algorithm Description
3. line number range
4. Step Count
5. Step Chain
6. Output Buffer

Step -

A step is made of:

1. Step Name
2. Step Description
3. line number range
4. Operation Count
5. Operation Chain
6. Notes

Operator -

An operator is made of:

1. Operation Name
2. Operation Description
3. line number
4. Inputs
5. Outputs

Thermel Emissions Dump

File Type Name: .ted, thermal emissions dump

Goal: Create a file format that can be used to store all the operations that occur in a defined algorithm.

A| Magic Number : 00T01E10D11 | 4 bytes
B| 8bit algo name size
C| algo name
D| 4 byte step count
E| 4 byte output size
F| outputbuffer
G| Step Size
H| Step
    I| operation count
    J| operation size
    K| operation

Thermal Debug Requirements

Goal: Pares a .ted file and generate an interactive replay of the algorithm in action

Requirements:

1. Take in x .ted files in as argument inputs
2. process each ted file into an algorithm stream
3. allow for running each stream in singles and together
4. allow for defining the number of bits to frame per operation and per step
5. Play through the stream relative to the output, and operations associated with it.
6. Add offset for streams running together
7. Allow streams to be inverted and synchronized via offset.