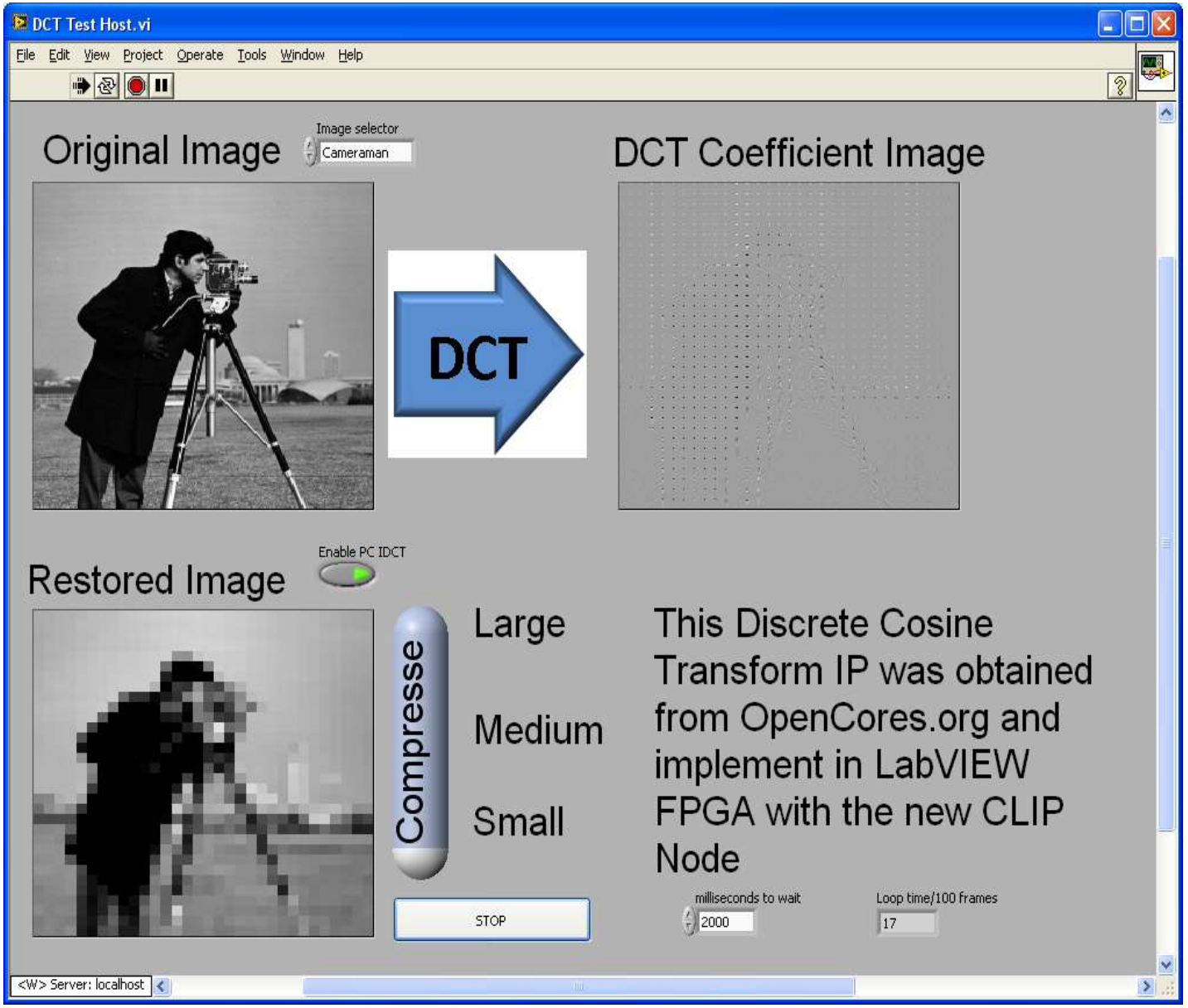
LabVIEW FPGA Experiences
Discrete Cosine Transform, DCT

NIDays09, Hørsholm Denmark
Civ. Ing. John Mølgaard, DELTA

Presentation overview

- Using VHDL in FPGA with CLIP
- NI example DCT CLIP
- Experiences with NI CLIP example
- NI DCT CLIP example modified
- DCT introduction
- Designing DCT/IDCT algorithms with LabVIEW
- Comparison of the two methods

Using VHDL in FPGA with CLIP



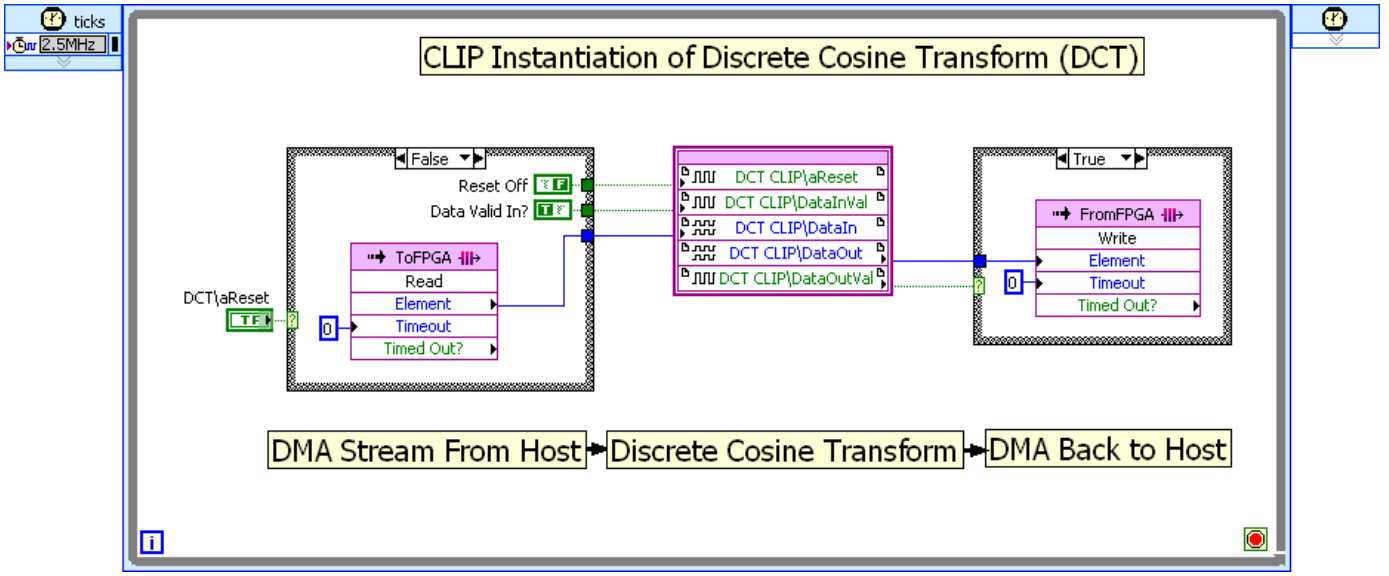
Importing External IP into LabVIEW FPGA with the CLIP Node

1. Create or acquire the IP
2. Define the interface to the IP using a declaration XML file (Try the CLIP XML Wizard)
3. Declare CLIP in the properties of an FPGA target
4. Add a CLIP item to a LabVIEW Project
5. Pass data between CLIP and an FPGA VI

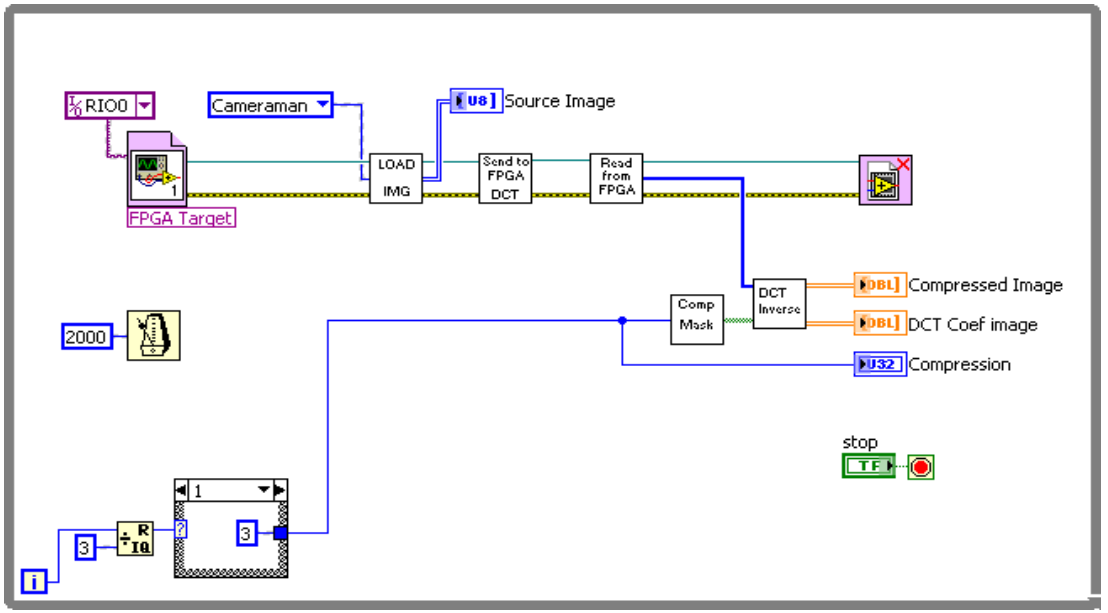
NI example DCT CLIP

- Download and unzip example
- Create new FPGA target (to your target)
- Select Clip Declaration file (XML)
- Create new 2.5M FPGA clock
- Add CLIP to FPGA target
- Drag FIFOs and Vi to new FPGA target
- Compile FPGA ? - No first reinstall CLIP
- Configure Open FPGA .vi reference

Example Block Diagrams



FPGA

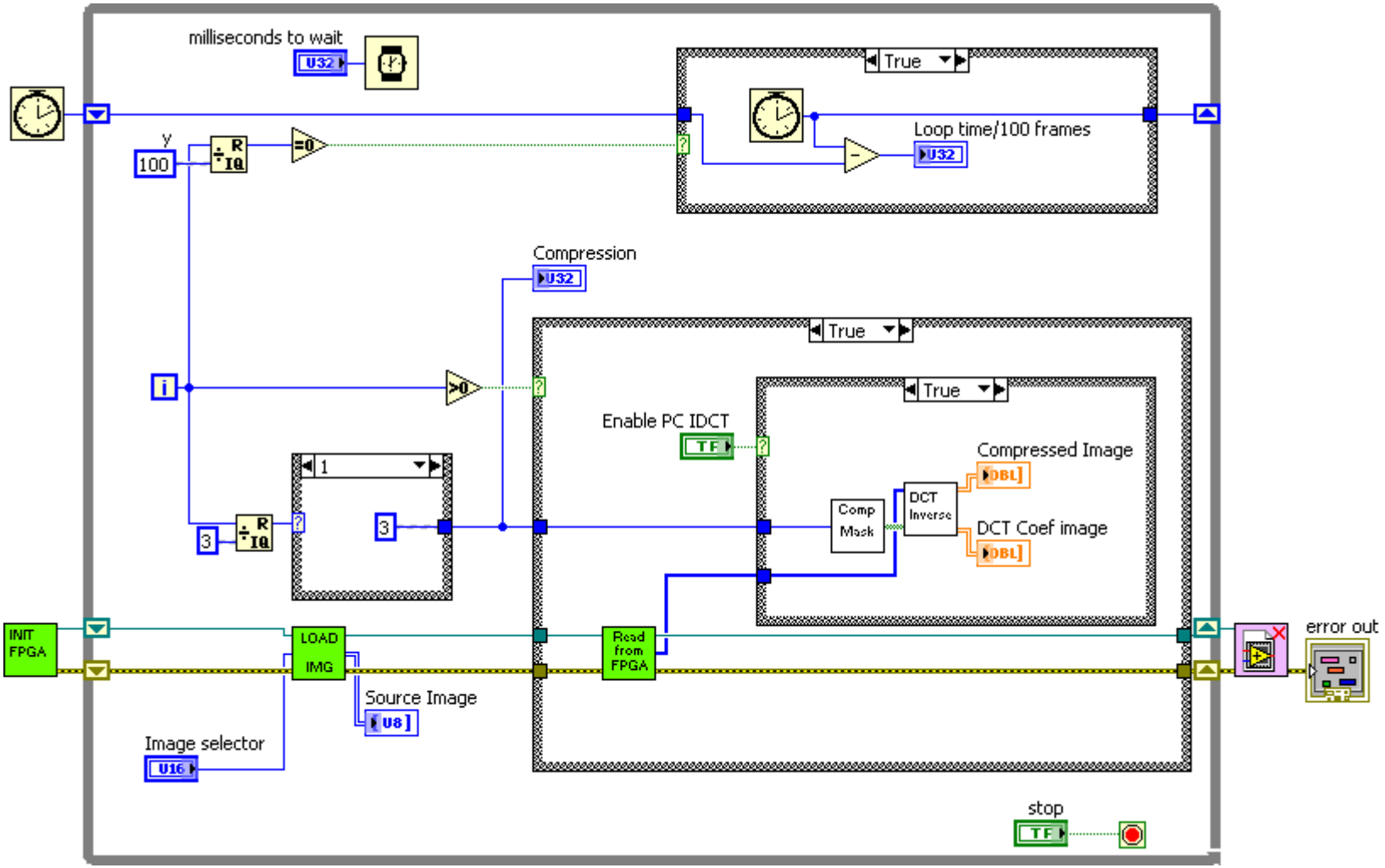


Host PC

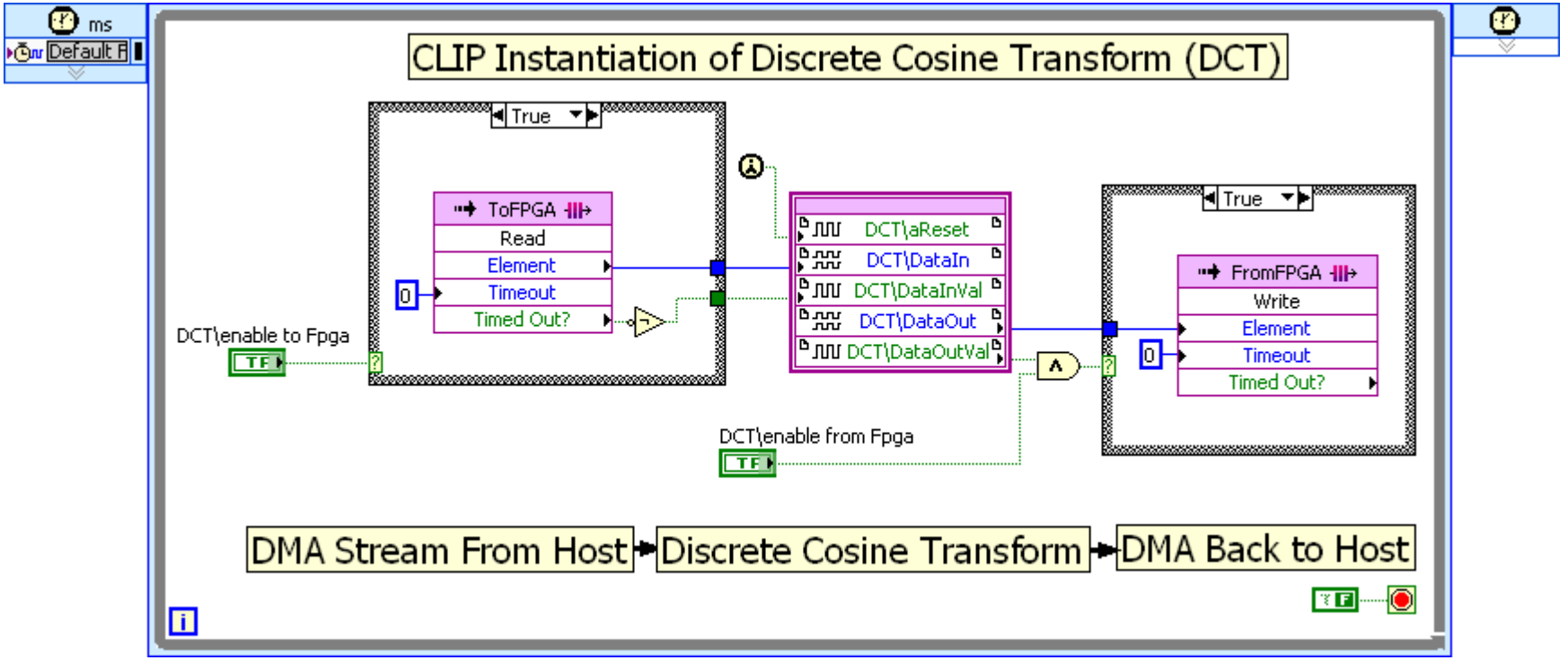
Example properties

- Demonstrates function, but not for practical use.
  - Reload FPGA for every run (aprox. 2 s)
  - Uses 2,5MHz clock
  - IDCT performed on host PC

Example revised, Host



Example revised , FPGA



Revised example properties

- Demonstrates function, for practical use.
  - Run continuously
  - Uses 40 MHz clock; Single Cycle Timed Loop
  - IDCT performed on host PC

DCT, Discrete Cosine Transform

- Used for image compression
- 2D transformation DCT – IDCT, similar to FFT
- Applied as 8*8 pixel tiles
- Image Intensity typically 8 bit representation

DCT Definition

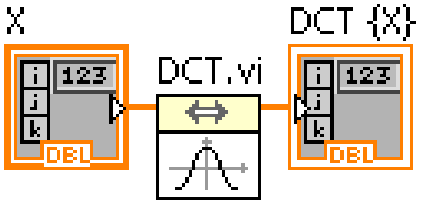
Definition:

$$\alpha(u) = \begin{cases} \sqrt{\frac{1}{N}} & for \quad u = 0 \\ \sqrt{\frac{2}{N}} & for \quad u \neq 0. \end{cases}$$

$$f(x,y) = \sum_{u=0}^{N-1}\sum_{v=0}^{N-1} \alpha(u)\alpha(v)C(u,v)\cos\left[\frac{\pi(2x+1)u}{2N}\right]\cos\left[\frac{\pi(2y+1)v}{2N}\right],$$

$$C(u,v) = \alpha(u)\alpha(v)\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f(x,y)\cos\left[\frac{\pi(2x+1)u}{2N}\right]\cos\left[\frac{\pi(2y+1)v}{2N}\right],$$

LabVIEW does it easily:



But not for FPGA!

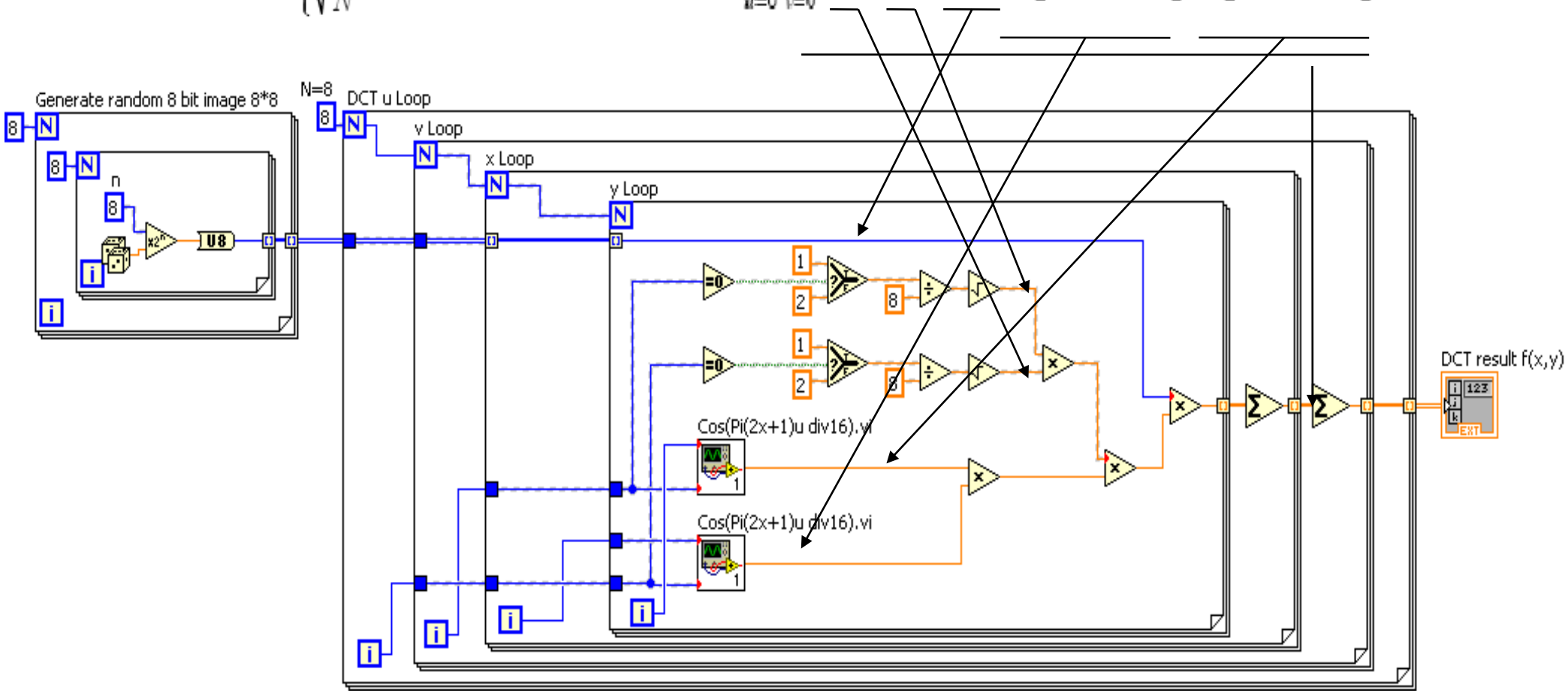
Strategy for developing FPGA DCT

- Implement DCT in LV from definition
- Restrictions only 8 bit and size 8*8
- Look for efficient implementations
- Transform implementation to legal FPGA code
- Test it compared to DCT.vi

Implement DCT in LV from definition

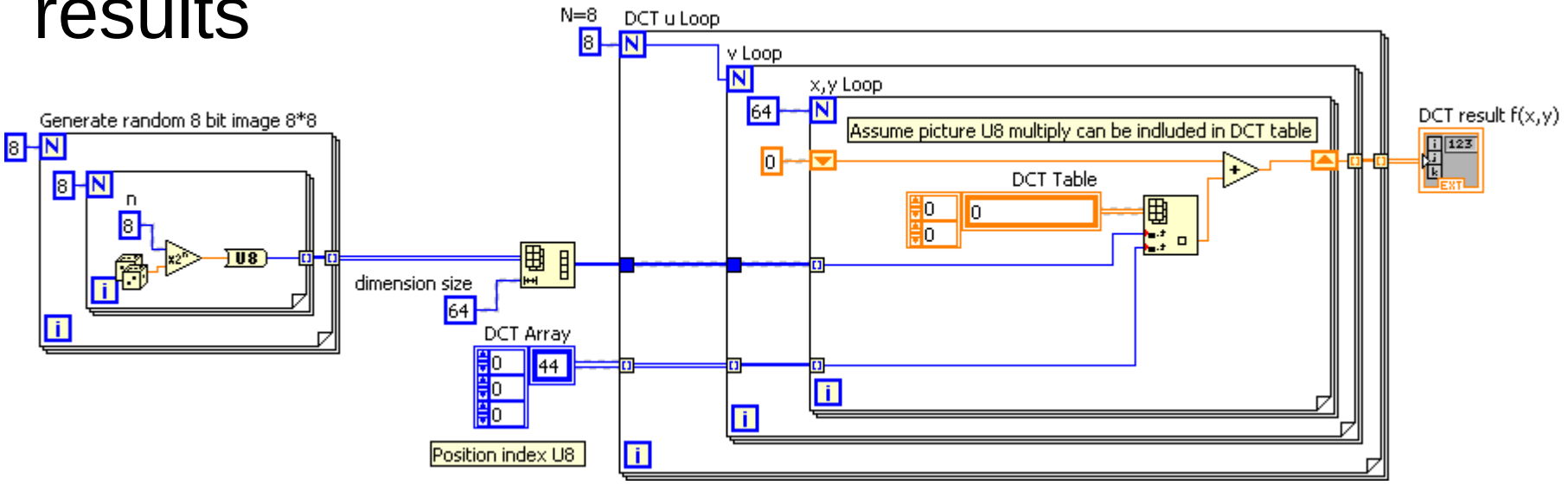
$$\alpha(u) = \begin{cases} \sqrt{\frac{1}{N}} & for \quad u = 0 \\ \sqrt{\frac{2}{N}} & for \quad u \neq 0. \end{cases}$$

$$f(x,y) = \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} \alpha(u)\alpha(v)C(u,v)\cos\left[\frac{\pi(2x+1)u}{2N}\right]\cos\left[\frac{\pi(2y+1)v}{2N}\right],$$



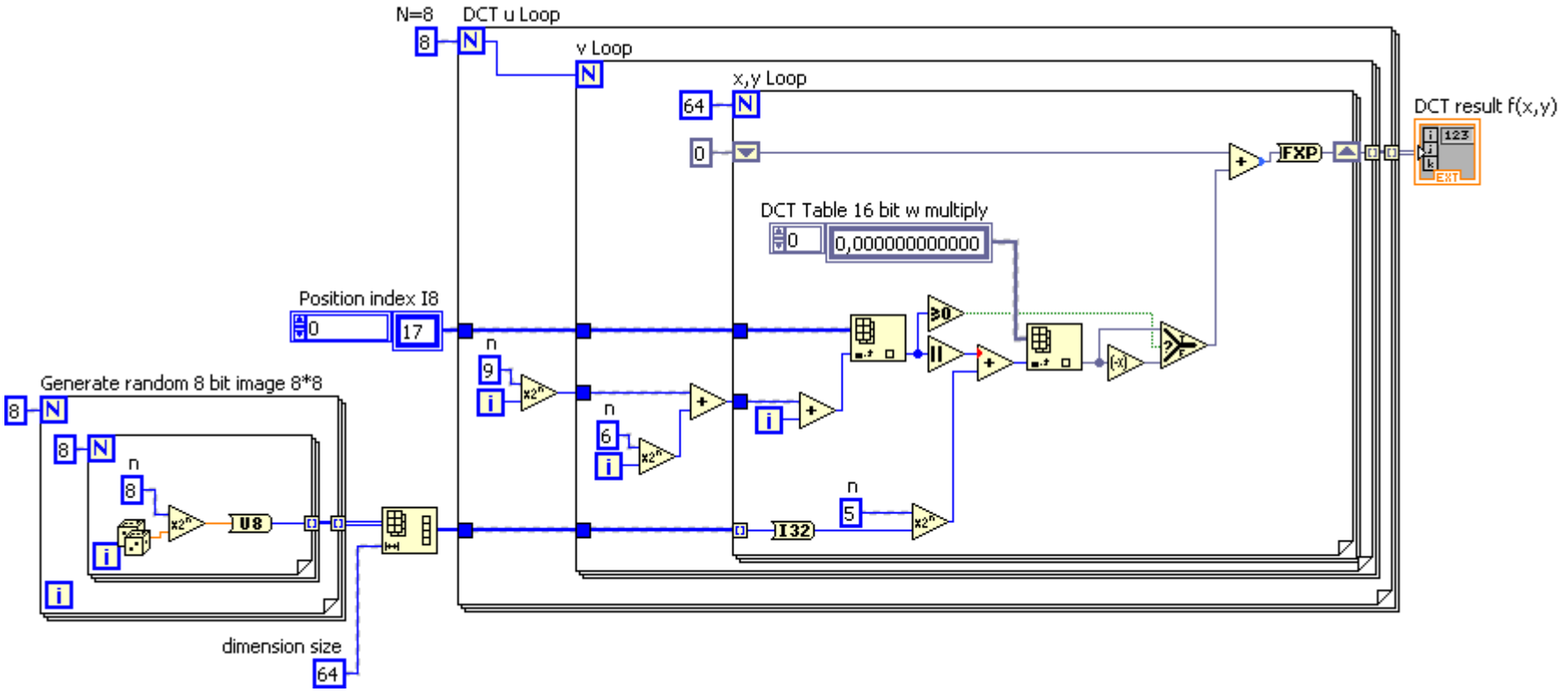
Look for efficient implementations

- a(u)a(v)cos(Pi(2x+1)u/2N)cos(Pi(2y+1)v/2N) only assumes 56 different values
- 8 bit is only 256 different values, so we can only have 56*256=14336 different multiplication results

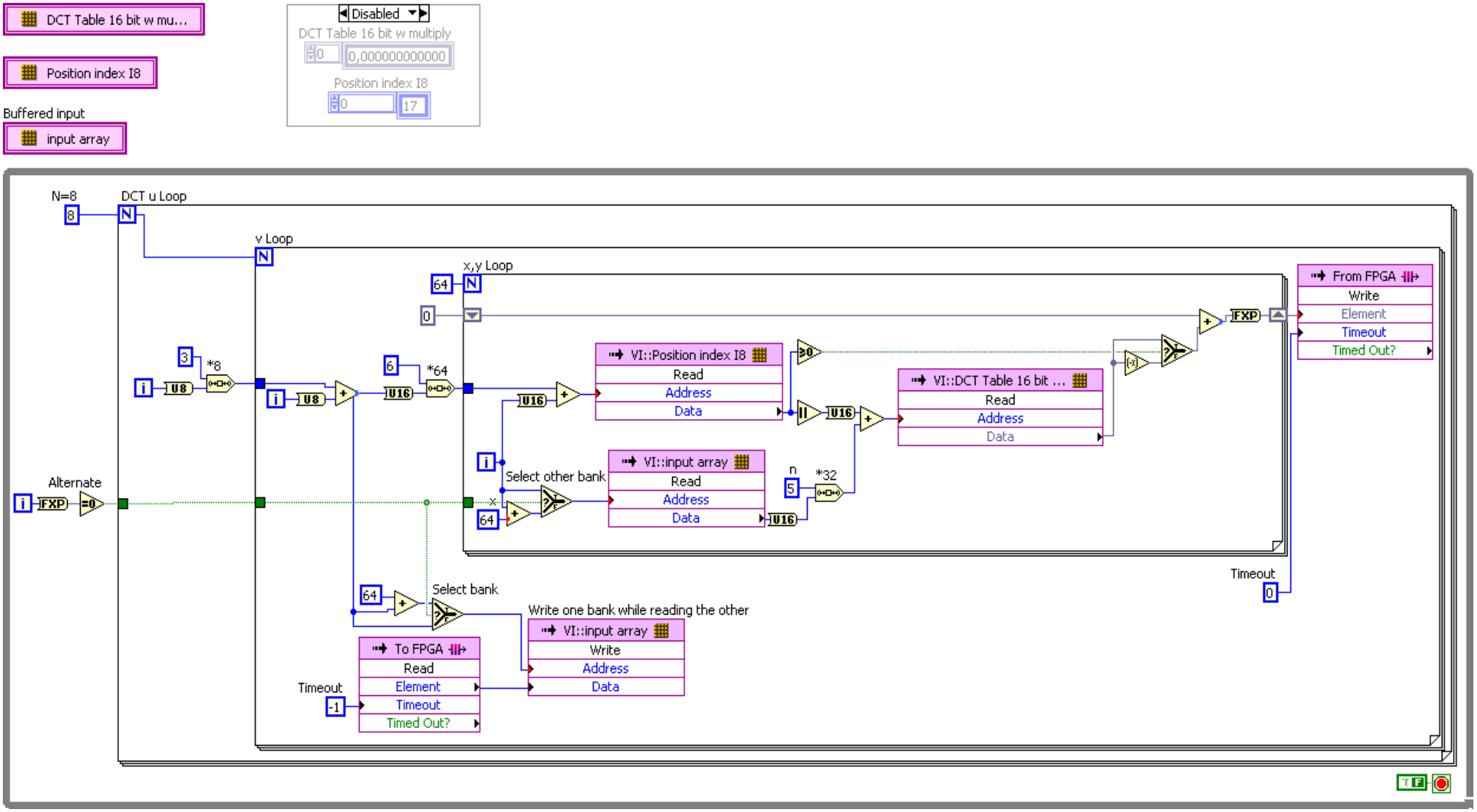


Look for efficient implementations 2

- Use FXP
- Halve DCT Table using symmetry +/-

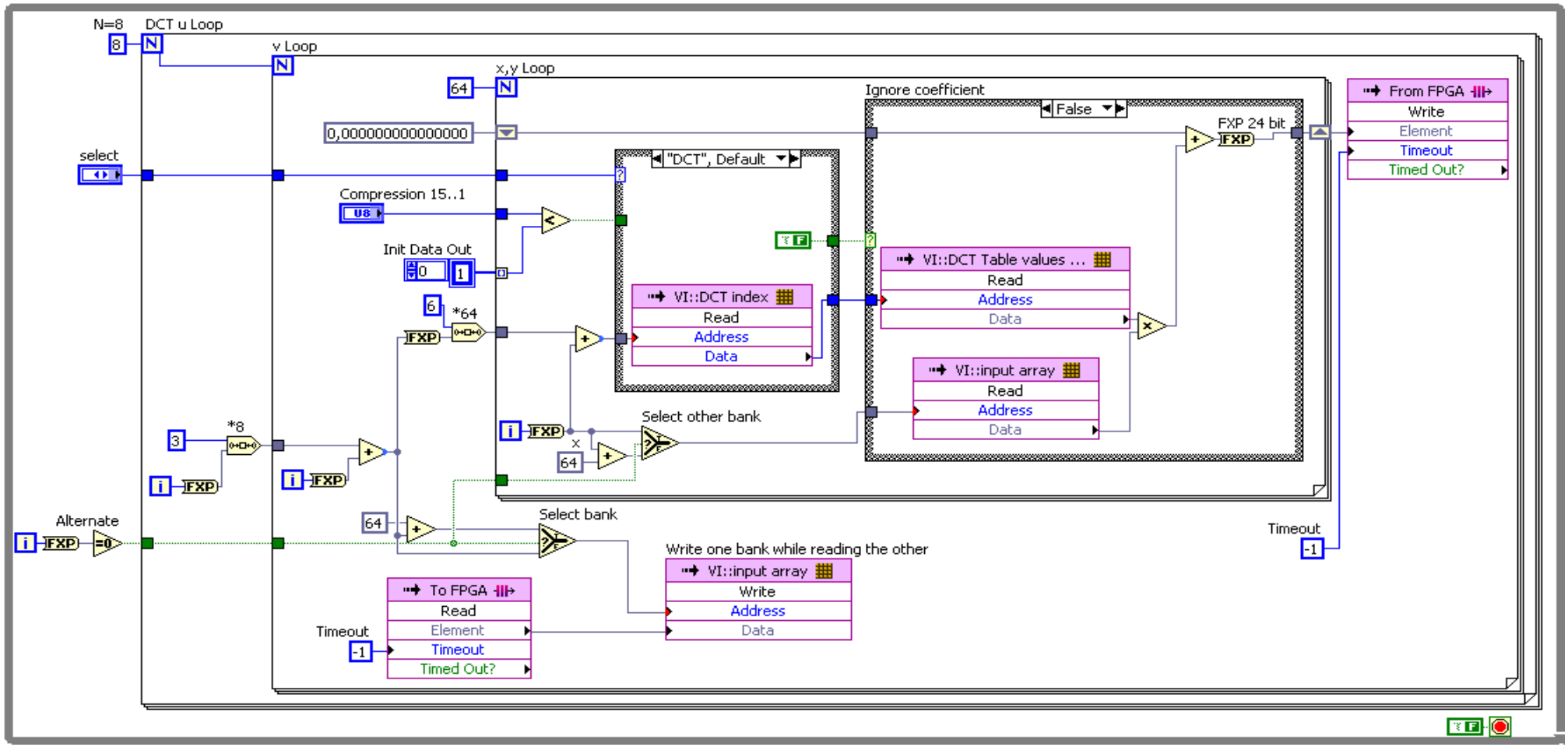


FPGA DCT U8 no Multiply



FPGA DCT/IDCT

Read only
DCT index
Read Only
IDCT index
Read Only
DCT Table values FXP ...
Buffered input
input array



Comparison of the two methods

- Fastest implementation is CLIP Revised
- FPGA Resources used:
  - NO CLIP means DMA to and from FPGA

%	CLIP	DCT no Multiply	DCT/IDCT	NO CLIP
Slice FF	6	4	6	3
4 input LUT	14	5	7	4
RAMB16s	20	19	18	18
SLICEs	19	8	10	6