JPEG Decoder

Saajan Singh Chana
6.375 Final Project
Spring 2008

Outline

- Overview of JPEG
- Problem scope
- Implementation
- Further work

JPEG Overview

- Developed in late 1980s (Joint Photographic Experts Group)
- Optimized for photorealistic (continuous-tone) images
- Not so good for text/computer-generated graphics
- Wide-ranging standard covering many possible applications

Baseline JPEG

- Most common implementation
- Design criteria:
  - Simple encoder/decoder implementation
  - Minimal memory usage
  - Suitable for most common use cases

Baseline JPEG

- 8 bits per pixel (per component)
- YCbCr colourspace
- Sequential encoding order
- Huffman used for entropy coding
- Maximum of 2 sets of Huffman tables
- All tables encoded in file

Encoding Algorithm

- Fundamental unit is 8x8 tile
- DCT
- Quantization
- Reording and entropy coding

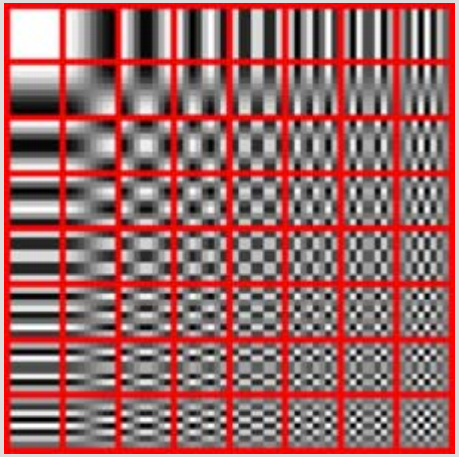
Discrete Cosine Transform

- Similar to DFT...but uses only real multiplications

$$f(x,y) = \frac{1}{4}\sum_{u=0}^{n-1}\sum_{v=0}^{n-1} C(u)C(v)A(x,u)A(y,v)F(u,v)$$

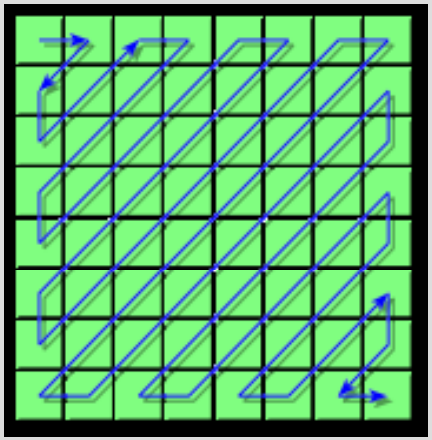
where

$$\begin{aligned} n &= 8 \\ C(u) &= \begin{cases} 1\sqrt{2} & u = 0 \\ 0 & \text{otherwise} \end{cases} \\ A(x,u) &= \cos\frac{(2x+1)u\pi}{16} \end{aligned}$$



Quantization and Reordering

- Quantization – element-by-element division by quantization matrix
  - Reduces bitdepth from 12 to 8, and correspondingly encoded size
  - Less important information discarded



- Diagonal reordering keeps coefficients of similar frequency together

Entropy coding

- Coefficients stored as variable-length integers (varints)
- Lower magnitude numbers take less space
- Each coefficient preceded by an 8-bit token:
  - Zero-run length
  - Varint size
  - (0, 0) is end of block
- Tokens are Huffman coded

Multicomponent Images

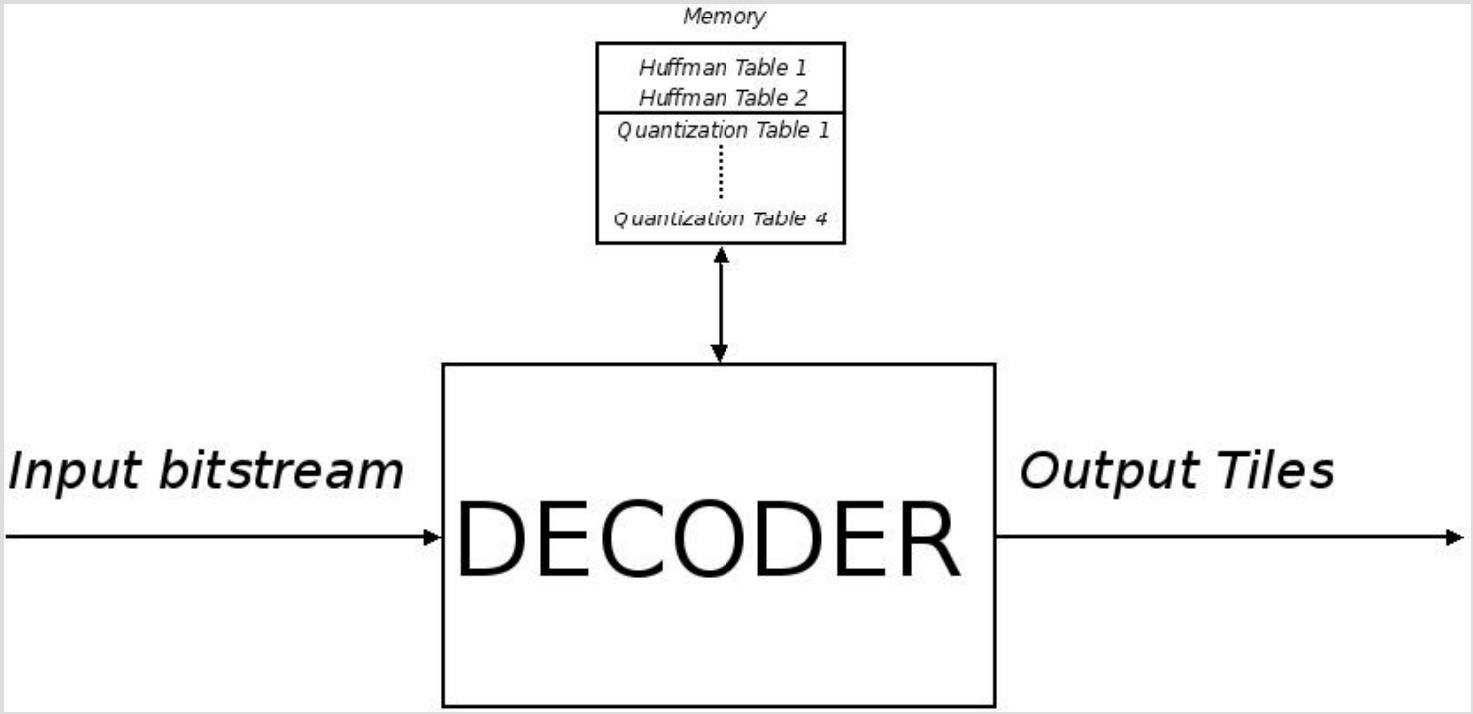
- Colour requires 3 components (Y, Cb, Cr)
- Chroma components can be lower resolution
- Each component can use different tables
- Interleaved: decoder must switch tables as required
- `Minimum Coded Block'
  - Up to 10 tiles

Outline

- Overview of JPEG
- Problem scope
- Implementation
- Further work

Problem Scope

- Bitstream decoder
- Not file decoding logic



Test plan

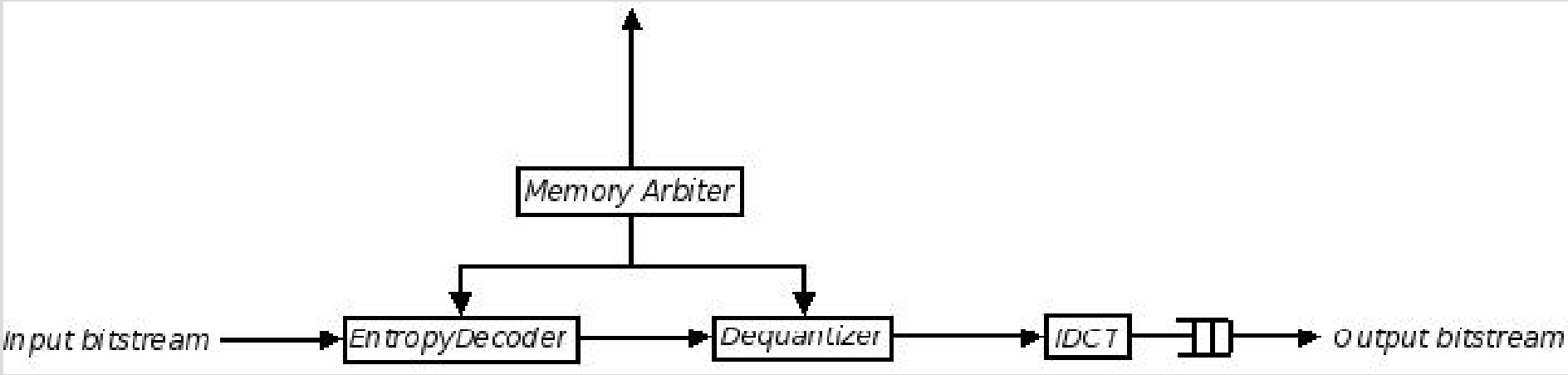
- C program to read in JFIF file and dump out data structures, image bits.
- Testbench sets up tables, streams image data in, displays output

Outline

- Overview of JPEG
- Problem scope
- Implementation
- Further work

Implementation

- Intermodule communication via Get/Put interfaces
- Memory access via Client interface



Huffman Decoding

- Tree structure requires many pointer dereferences
- `Easy' representation very sparse
- Instead, record highest valid codeword for each code length and compute offset into flat array
- Easy to generate from info in JPEG file
- Cutoffs/offsets cached to reduce memory accesses

Dequantizer & IDCT

- Fixed-point
- Uses standard Verilog multipliers
- IDCT is O(n^2)

Outline

- Overview of JPEG
- Problem scope
- Implementation
- Further work

Further Work

- Debugging... (!)
- Faster DCT implementation
- Experiment with multiplier design
- Trade off – size vs. memory accesses
- Optimize memory arbiter

Questions