Accelerating Bit Error Rate Simulation in MATLAB using Graphics Processors

High-Performance Embedded Computing Workshop (HPEC 2011)
22 September 2011

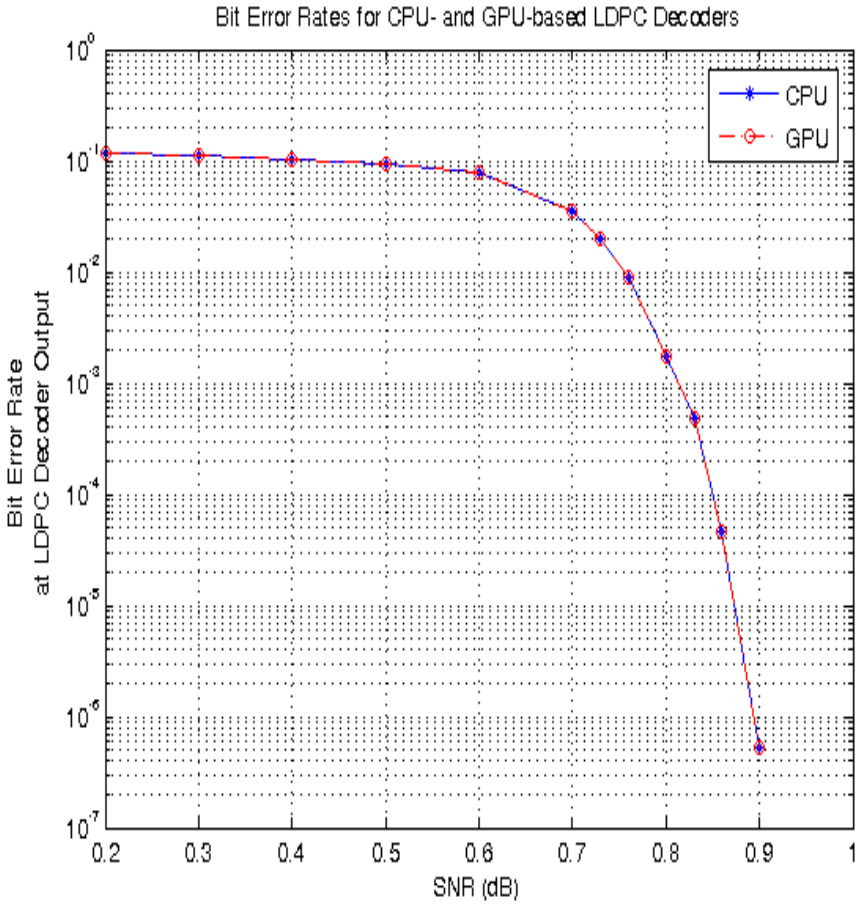
James Lebak
Brian Fanous
Nick Moore

Outline

- Application Area
- MATLAB Tools
- DVB-S.2 Subset benchmark
- DVB-S Subset benchmark
- Conclusions

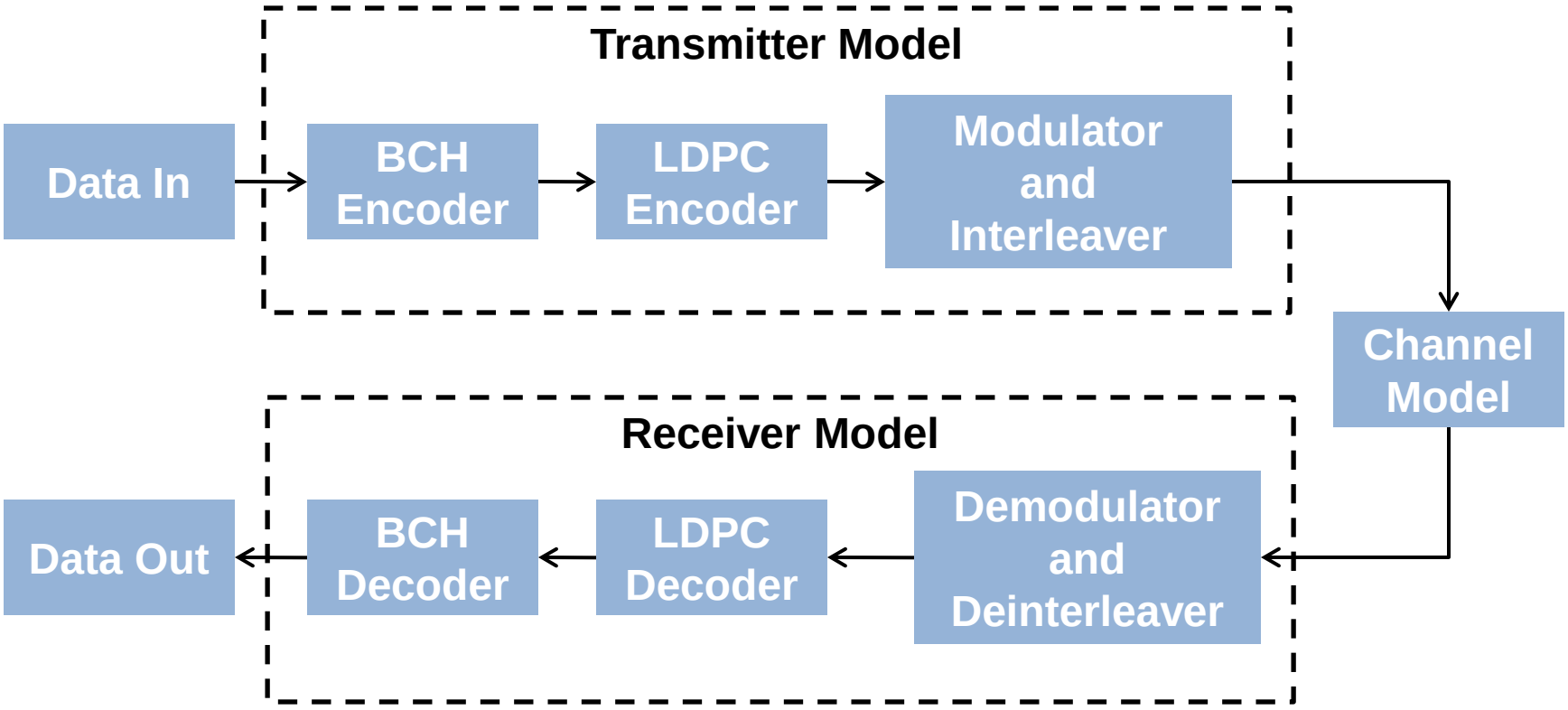
Bit Error Rate Simulation

- Begin with a model of a communications system
- Vary the signal-to-noise ratio
- Calculate bit error rate through Monte Carlo simulation
- Obtain a curve like that at the right
- Expensive (millions of trials, days of simulation)



Example System Model

Digital Video Broadcast Standard, Second Generation (DVB-S.2)



LDPC = Low-Density Parity Check

Simulation Goal: Verify transmitter and receiver achieve a given bit error rate in the presence of noise introduced by the channel model

Outline

- Application Area
- MATLAB Tools
- DVB-S.2 Subset benchmark
- DVB-S Subset benchmark
- Conclusions

System Objects

System objects allow design of dynamic systems in MATLAB

Represent dynamic systems

- Algorithms with input values and states that change over time

Optimized for iterative execution

- One-time checking of parameter values, input arguments
- Data streaming between objects

Many algorithms implemented in C++ for speed

Support for automatic C code generation

System Object Receiver Model

```
%***Receiver Object Creation ***
hDemod = comm.PSKDemodulator(demodulatorArgs{:});
hDeintrlv = comm.BlockDeinterleaver(dvb.InterleaveOrder);
hDec = comm.LDPCDecoder(decoderArgs{:});
```

System Object Receiver Model

```
%***Receiver Object Creation ***
hDemod = comm.PSKDemodulator(demodulatorArgs{:});
hDeintrlv = comm.BlockDei[illegible]aver(dvb.InterleaveOrder);
hDec = [illegible]
```

```
{'ModulationOrder', 2^dvb.BitsPerSymbol, ...
 'BitOutput', true, ...
 'PhaseOffset', dvb.PhaseOffset, ...
 'SymbolMapping', 'Custom', ...
 'CustomSymbolMapping', dvb.SymbolMapping, ...
 'DecisionMethod', 'Approximate log-likelihood ratio', ...
 'Variance', dvb.NoiseVar}
```

System Object Receiver Model

```
%***Receiver Object Creation ***
hDemod = comm.PSKDemodulator(demodulatorArgs{:});
hDeintrlv = comm.BlockDeinterleaver(dvb.InterleaveOrder);
hDec = comm.LDPCDecoder(decoderArgs{:});
```

```
%***Receiver Model Execution***%
demodOut    = step(hDemod, chanOut);
dexlvrOut   = step(hDeintrlv, demodOut);
ldpcdecOut  = step(hDec, dexlvrOut);
```

Graphics Processing Unit (GPU) Acceleration in MATLAB

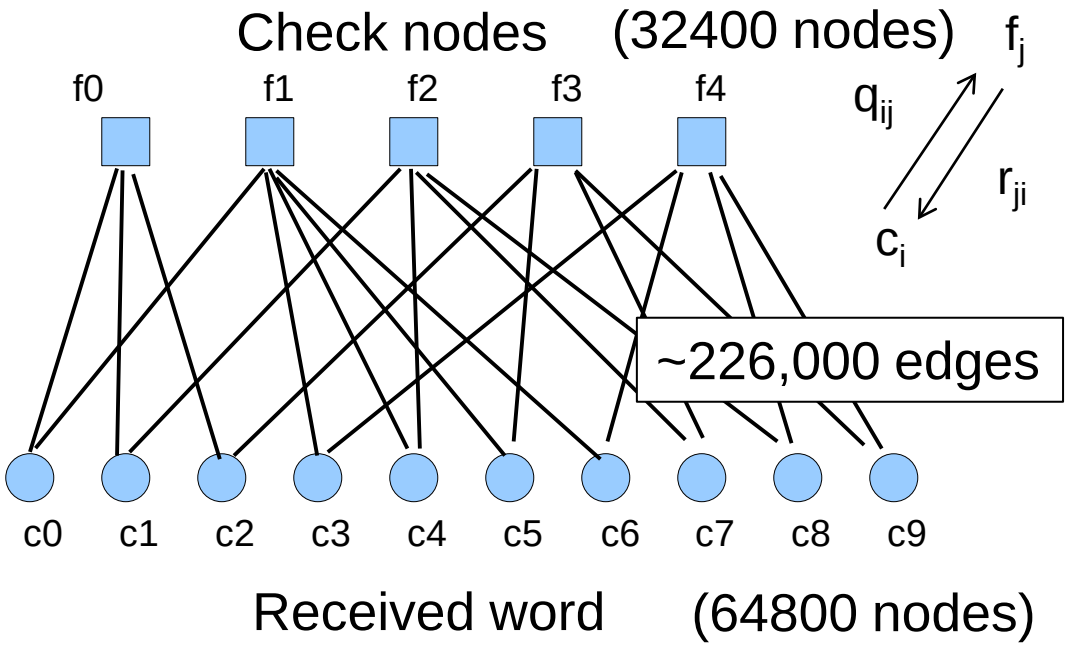
Level of control	*Required effort*	*Parallel options*
Minimal	**None**	**Use GPU arrays with MATLAB built-in functions**
Some	**Straightforward**	**Execute custom functions on elements of GPU arrays**
Extensive	**Involved**	**Create kernels from existing CUDA code and PTX files**

Requirements

- **Parallel Computing Toolbox**
- **NVIDIA GPU with Compute Capability 1.3 or greater**

LDPC Decoder

LDPC Decoder was implemented using custom CUDA kernels in MATLAB release R2011a



Iteratively determine message bits that best satisfy system constraints

LDPC Decoder Algorithm

```
for ii=1:maxIterations,
        Update all r messages (in parallel)
        Update all q messages (in parallel)
end
form hard/soft decision (in parallel)
```

GPU-Accelerated Receiver Model

```
%***Receiver Object Creation ***
hDemod = comm.PSKDemodulator(demodulatorArgs{:});
hDeintrlv = comm.BlockDeinterleaver(dvb.InterleaveOrder);
hDec = comm.gpu.LDPCDecoder(decoderArgs{:});

  %***Receiver Model Execution***%
  demodOut    = step(hDemod, chanOut);
  dexlvrOut   = step(hDeintrlv, demodOut);
  ldpcdecOut  = step(hDec, dexlvrOut);
```

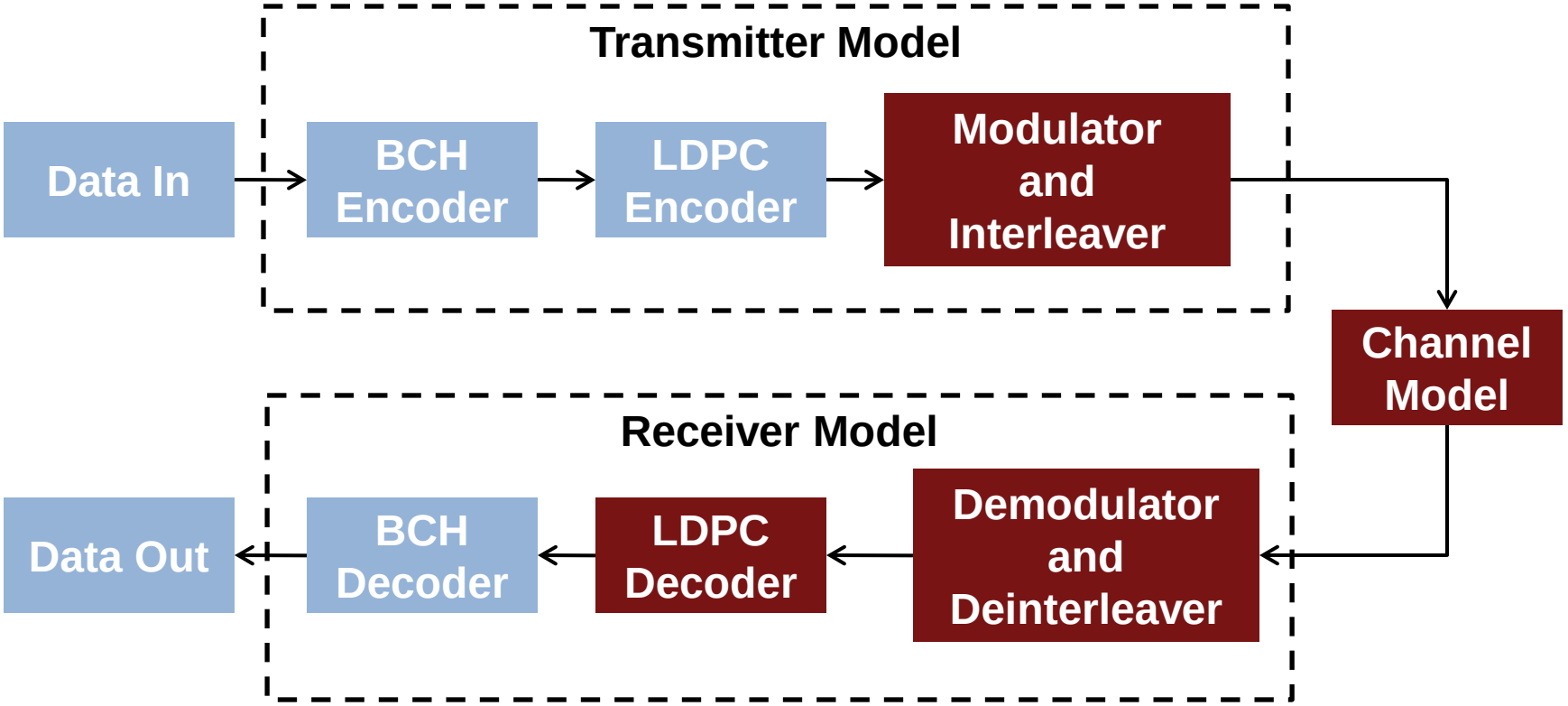
- Minimal code changes to use GPU System Object
- GPU System object is 20x faster than original System object
- Overall system simulation is 5x faster

Outline

- Application Area
- MATLAB Tools
- DVB-S.2 Subset benchmark
- DVB-S Subset benchmark
- Conclusions

DVB-S.2 Subset Benchmark

Digital Video Broadcast Standard, Second Generation (DVB-S.2)



Subset benchmark (can execute entirely on GPU)
New objects implemented using MATLAB for faster development

Using GPU System objects

- GPU System objects are *drop-in replacements* for standard System objects
  - Normal MATLAB arrays in ➔ normal MATLAB arrays out
  - Requires two data transfers

Using GPU System objects

- GPU System objects are *drop-in replacements* for standard System objects
  - Normal MATLAB arrays in ➔ normal MATLAB arrays out
  - Requires two data transfers

```
%***Receiver Object Creation ***
hDemod = comm.gpu.PSKDemodulator(demodulatorArgs{:});
hDeintrlv = comm.gpu.BlockDeinterleaver(dvb.InterleaveOrder);
hDec = comm.gpu.LDPCDecoder(decoderArgs{:});

  %***Receiver Model Execution***%
  demodOut    = step(hDemod, chanOut);
  dexlvrOut   = step(hDeintrlv, demodOut);
  ldpcdecOut  = step(hDec, dexlvrOut);
```

Using GPU System objects

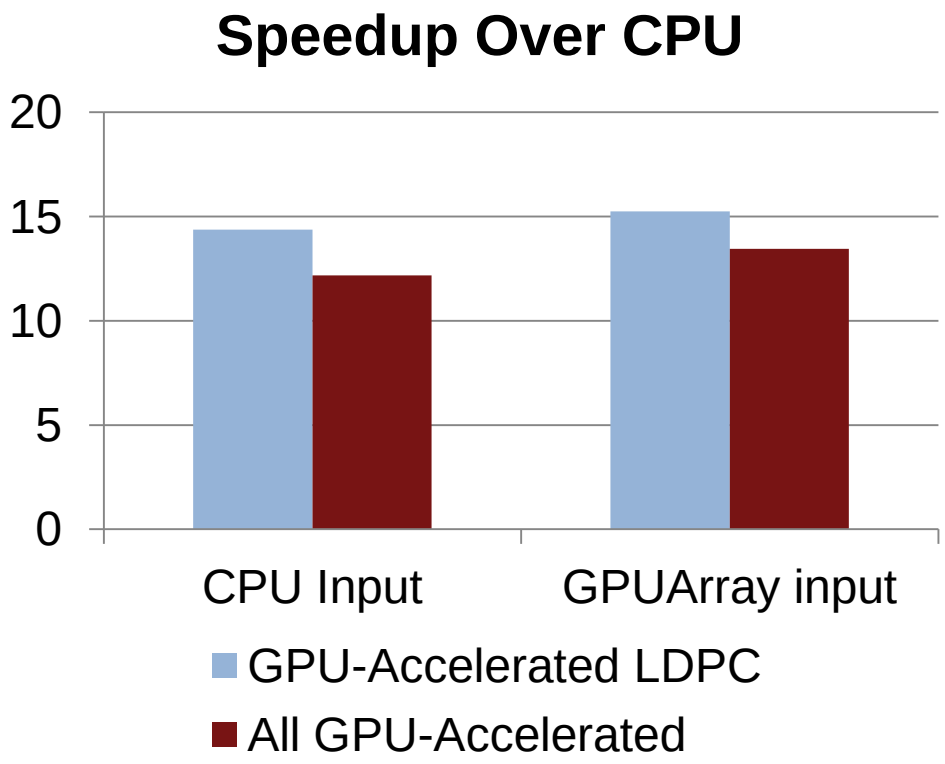
- GPU System objects are *drop-in replacements* for standard System objects
  - Normal MATLAB arrays in ➔ normal MATLAB arrays out
  - Requires two data transfers
- GPU System objects also accept GPUArrays as inputs
  - In this case the output is also a GPU Array
  - Avoids data transfer overhead

```
%***Receiver Object Creation ***
hDemod = comm.gpu.PSKDemodulator(demodulatorArgs{:});
hDeintrlv = comm.gpu.BlockDeinterleaver(dvb.InterleaveOrder);
hDec = comm.gpu.LDPCDecoder(decoderArgs{:});

  %***Receiver Model Execution***%
  demodOut    = step(hDemod, gpuArray(chanOut));
  dexlvrOut   = step(hDeintrlv, demodOut);
  ldpcdecOut  = gather(step(hDec, dexlvrOut));
```

DVB-S.2 Subset Benchmarks Using GPUArray

- **Baseline: speed of subset with all objects on the CPU**
- **Using GPUArray objects as inputs accelerates both versions of the benchmark**
- **Using the GPU-accelerated LDPC provides a large performance boost**
- **Adding GPU-accelerated versions of other objects slows down the simulation**

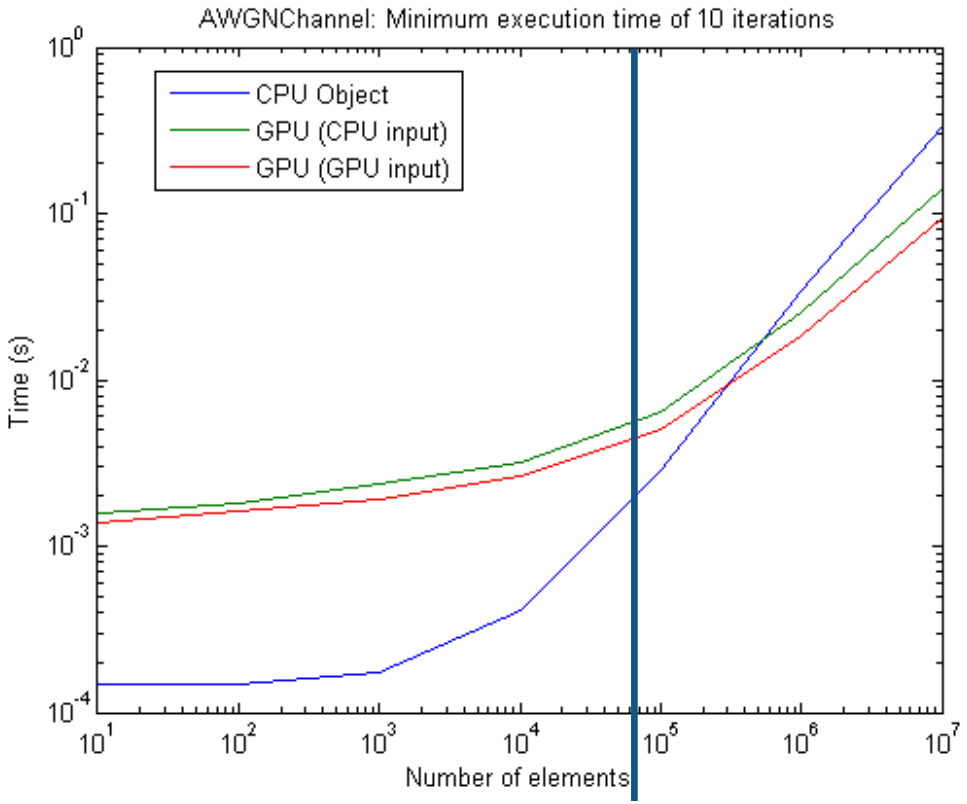


Benchmark Machine Specs

CPU: 2.5 GHz Intel Core 2 Quad
GPU: NVIDIA Tesla C1060
Windows 7
MATLAB R2011b

Performance vs. Input Size for GPU System Objects

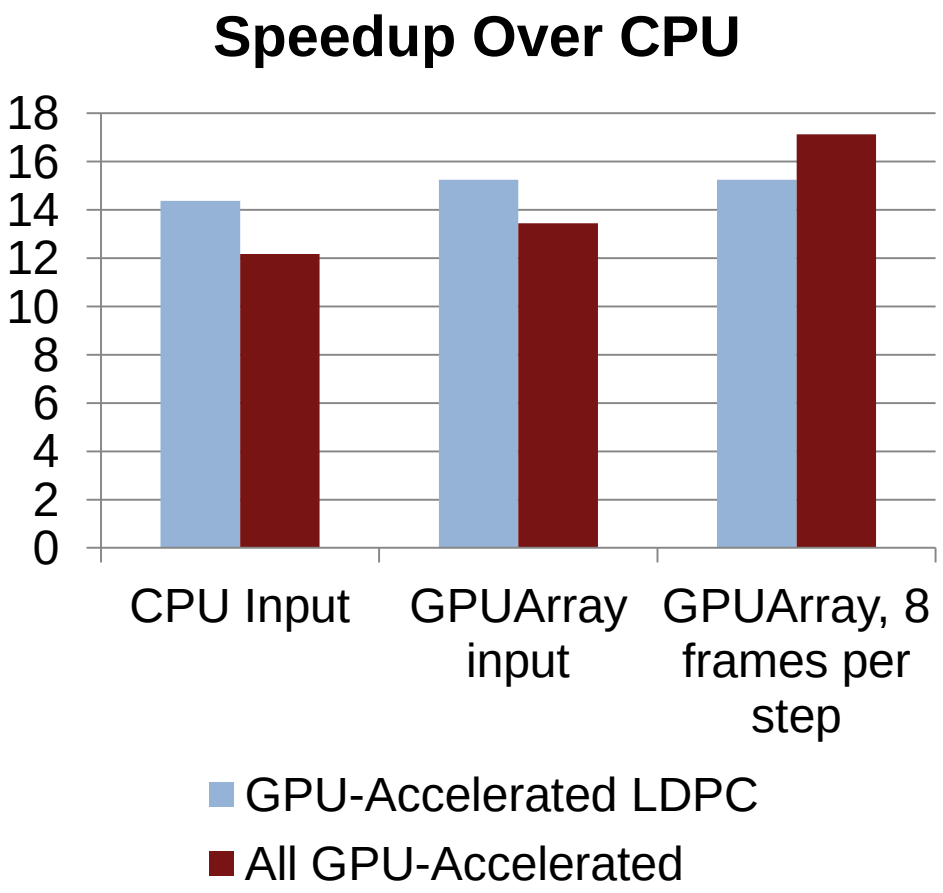
- **Computational density is important for obtaining good GPU performance**
- **Many GPU System objects are slower for small data sizes**
- **In R2011b, crossover point is generally around 10^5**
- **Frame size is 64800 elements**



System Frame Size

DVB-S.2 Subset Multi-Frame Benchmarks

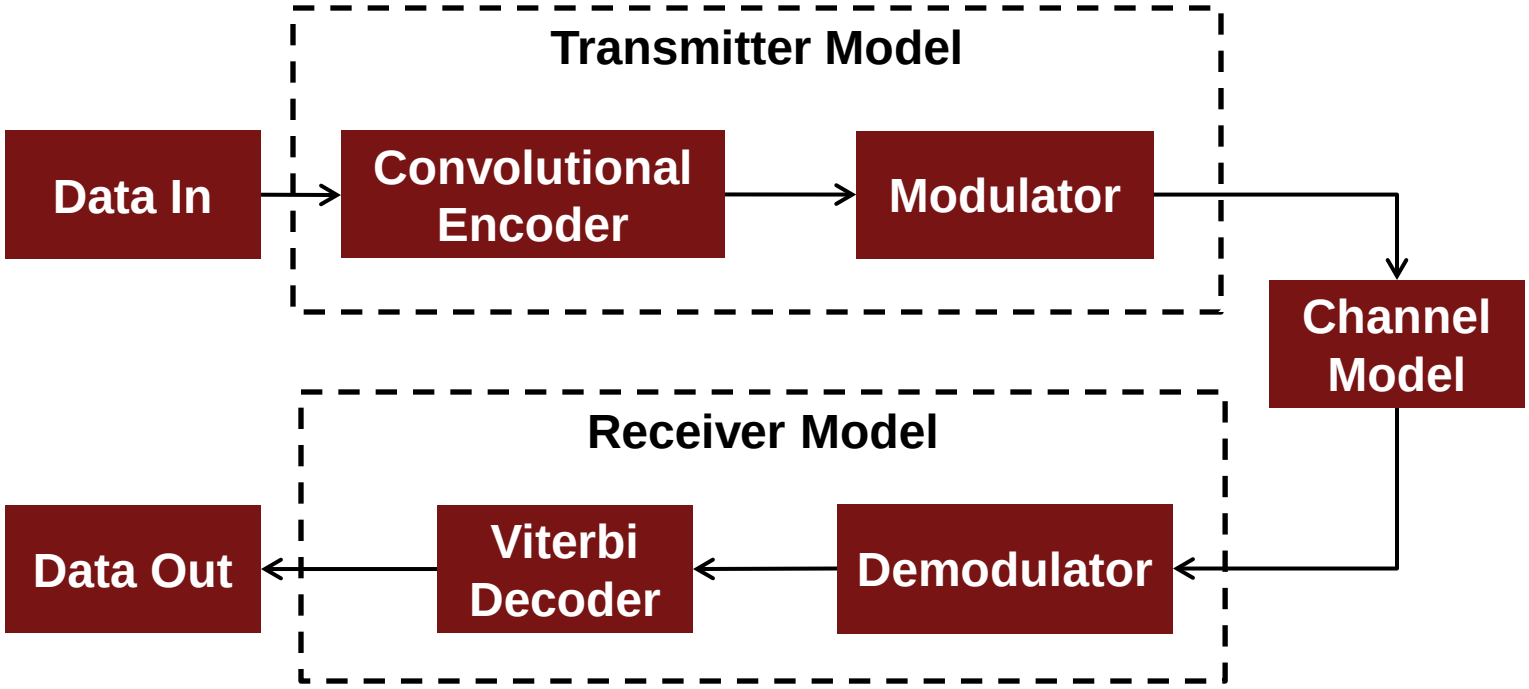
- **Process 8 frames of data on the GPU simultaneously to get better performance**
- **LDPC performance begins to saturate, but other objects become faster**



Outline

- Application Area
- MATLAB Tools
- DVB-S.2 Subset benchmark
- DVB-S Subset benchmark
- Conclusions

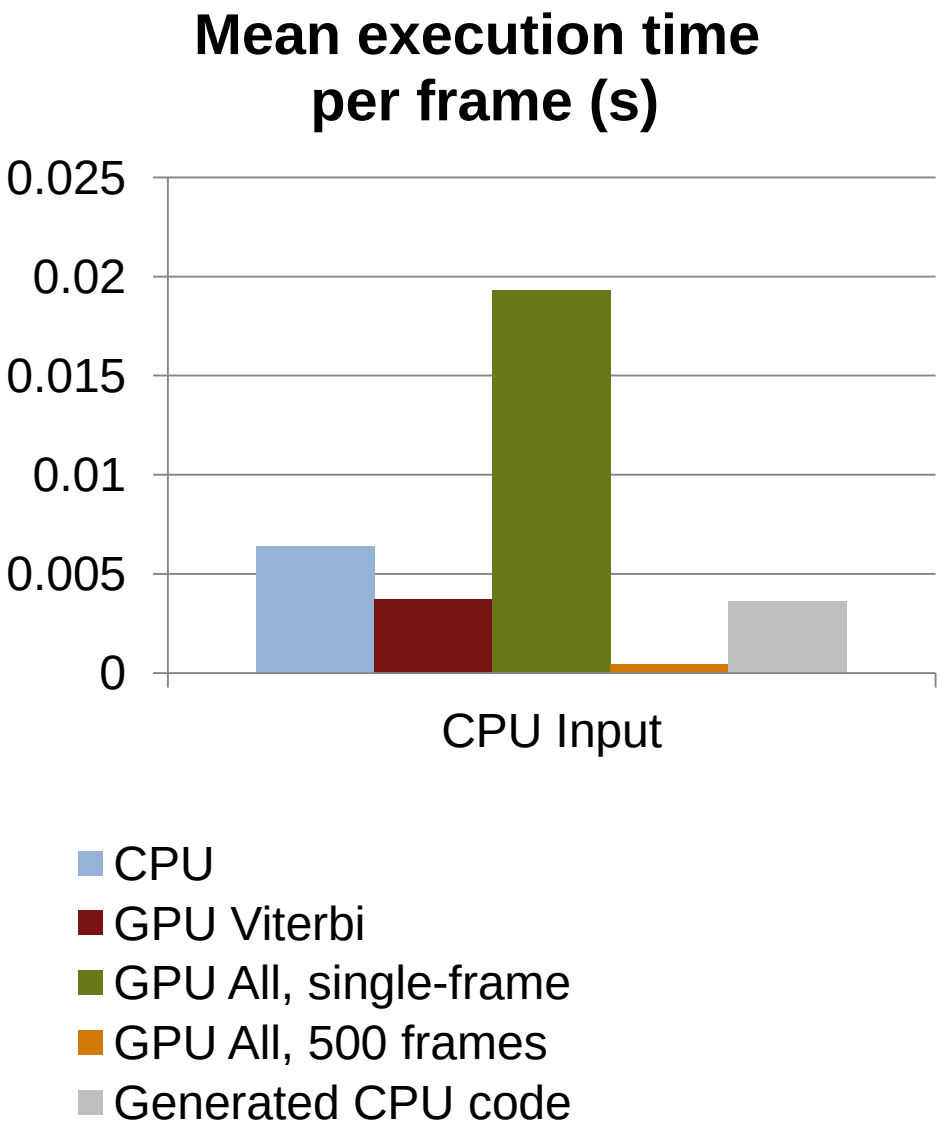
Digital Video Broadcast – Satellite (DVB-S) benchmark



- DVB-S is a similar system
  - uses different encoder/decoder pair than DVB-S.2
- The entire chain executes on the GPU
  - including message generation and validation
- Viterbi Decoder is implemented as a custom CUDA kernel
  - all others implemented using MATLAB GPU support

DVB-S Subset Benchmarks

- **DVB-S frame size is 1632 elements**
  - **single-frame performance on the GPU is slower than the CPU**
  - **multi-frame performance is much faster**



Conclusions

- The GPU can be a powerful tool for accelerating Bit Error Rate simulation in MATLAB
- Parallel Computing Toolbox allows a trade-off between work and performance
  - Implement the kernels with highest computational requirements as custom CUDA kernels
  - Implement kernels with lower computational requirements using MATLAB
- For best performance using the GPU, process multiple data frames simultaneously

Acknowledgments

The following people at MathWorks contributed to this work or to the presentation.

Andy Grace

Witek Jachimczyk

Amit Kansal

Darel Linebarger

Mike Longfritz

Roy Lurie

Mike McLernon

Don Orofino

Paul Pacheco

Narfi Stefansson

Bharath Venkataraman