High-rate Structural Health Monitoring: Part-II Embedded System Design

Austin Downey[1,2], Jason Bakos[3]

[1]Department of Mechanical Engineering
[2]Department of Civil and Environmental Engineering
[3]Department of Computer Science and Engineering
University of South Carolina, Columbia, SC 29208

UNIVERSITY OF SOUTH CAROLINA

Structures Experiencing High-Rate Dynamic Events

Applications:

1. Vehicle collision
2. Blast mitigation
3. Ballistic packages
4. Hypersonic vehicles
5. Hard Target Penetrating Weapons

Vehicle Collision

Active Blast Mitigation

Ballistics Packages

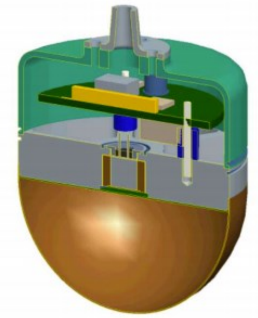
Hypersonic Vehicles

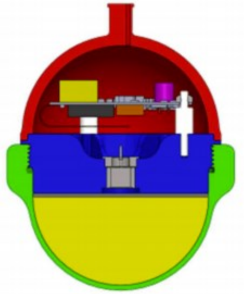
Hard Target Penetrating Weapons

Next Generation Fuzes

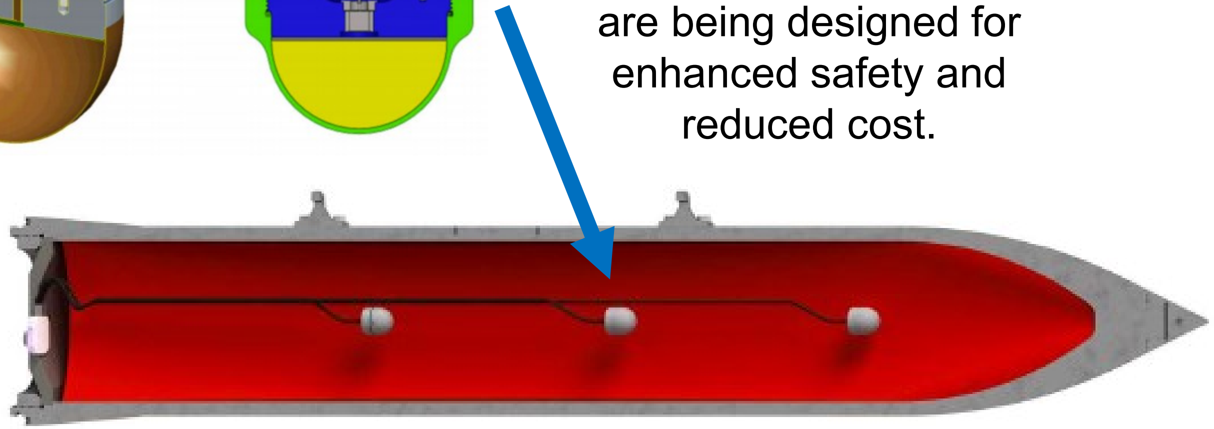
Thorough collaborations with the AFRL we are working on enabling technology for

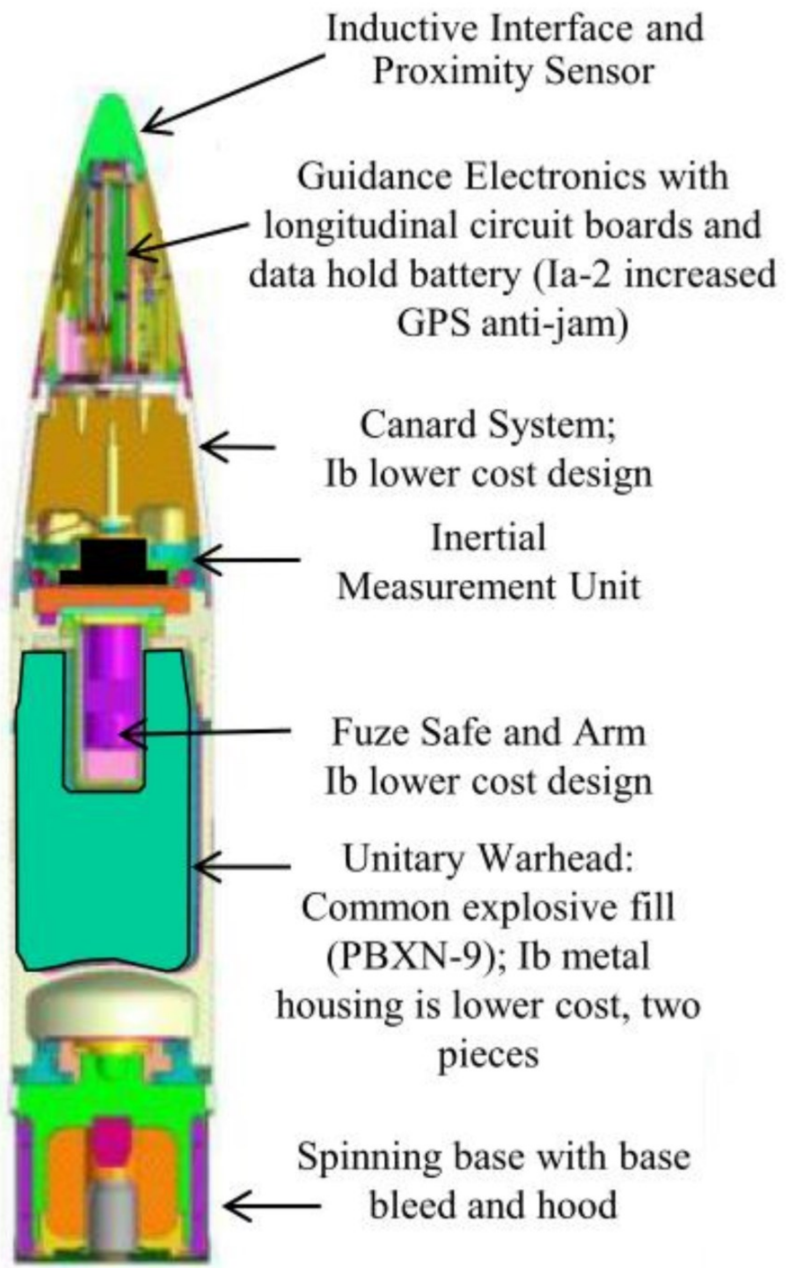
- Fuzes with decision-making capabilities
- Fuzes that can “adapt” to their condition
- Fuzes that are resilient to impact (e.g. after an impact, the are just as strong as before).





Embedded fuze design are being designed for enhanced safety and reduced cost.

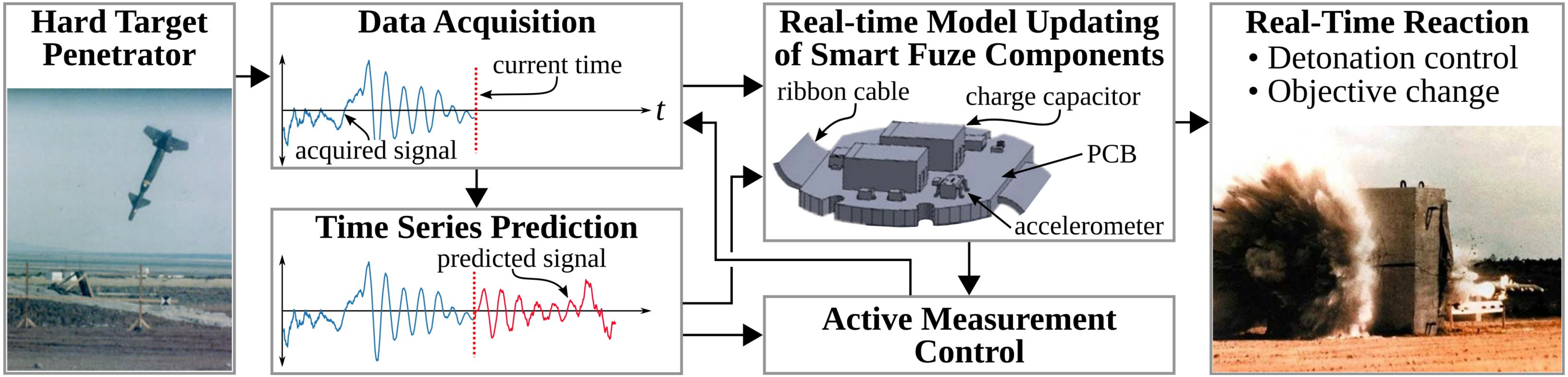




SAND2019-5293C

Next Generation Fuzes

Develop a computationally efficient real-time model-updating framework for structures experiencing high-rate dynamics capable of being executed on edge-computing platforms with a timestep of 1 ms

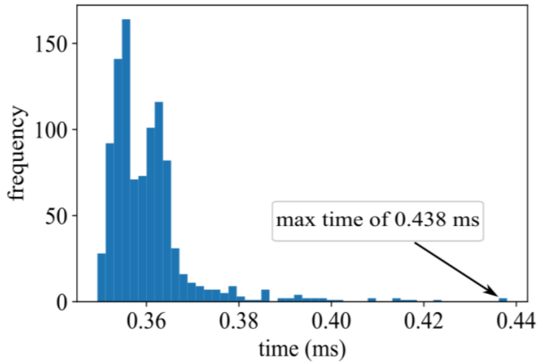


Key Challenges Identified

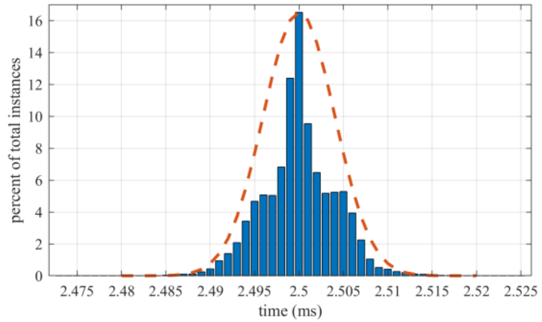
Key challenges:

1. The deterministic transfer of knowledge, and consequences of missing information, between data-driven models and controllers/decision-makers when such strict latencies are required.
2. The stability and robustness of model-updating schemes with short time steps, particularly an understanding of how faulty sensors affect the updated model.
3. The validation of low-latency real-time machine learning control schemes that are co-designed with hardware. Accurately capturing the effects of delays caused by data-acquisition systems and the complex interaction between controllers and the real-world plant is important to the controller's validation.

General Purpose Operating System (GPOS)



Real-time Operating System (RTOS)



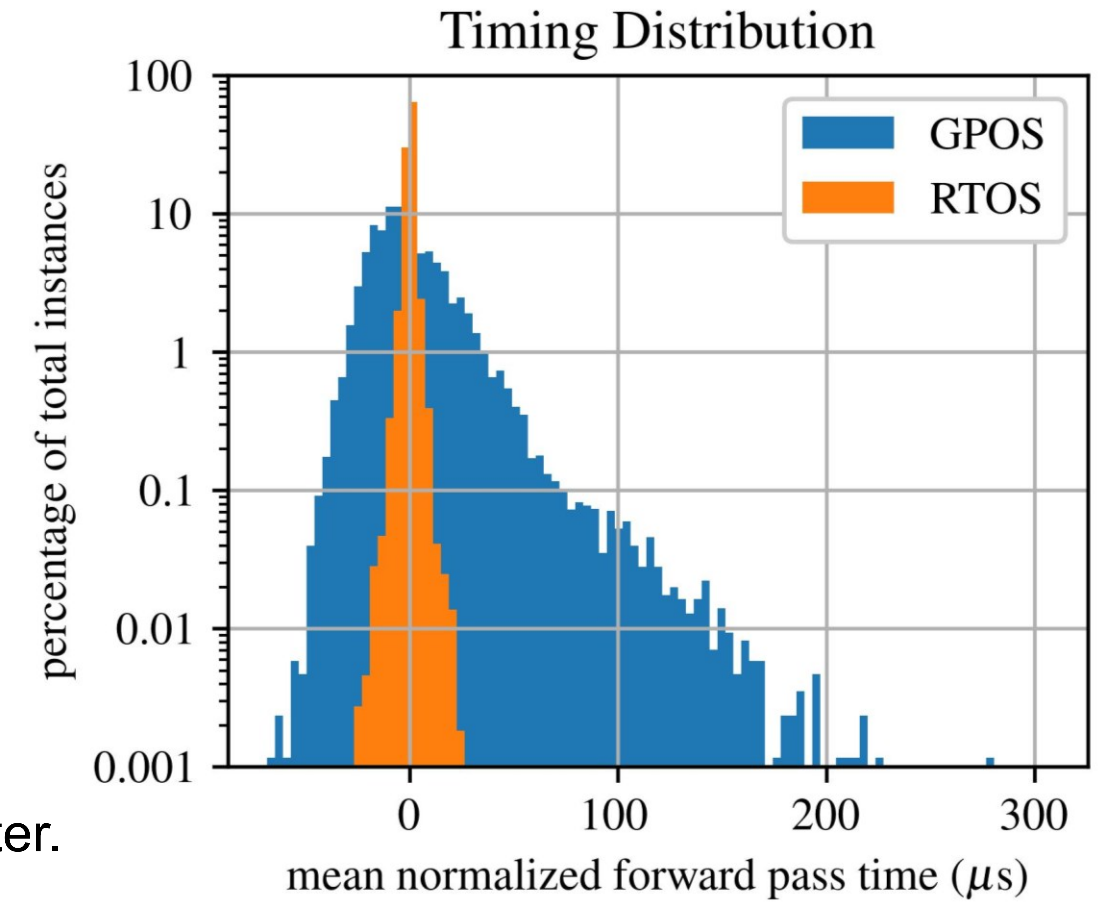
Blast Mitigation | LINE-X

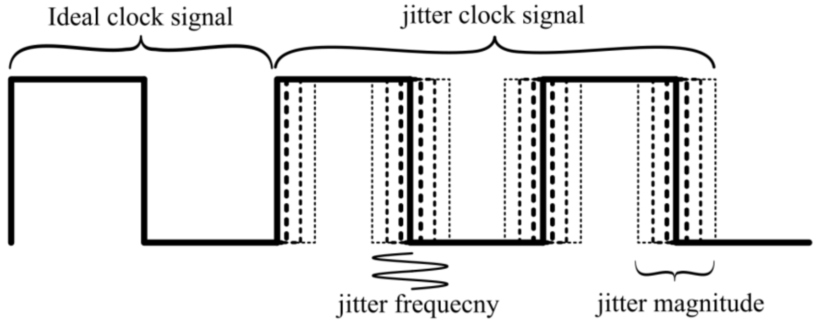
Hardware-software Co-design

Selecting Proper Hardware

A few hardware considerations:

- General Purpose Operating System (GPOS):
  - Easy to program.
  - No guarantees on timing, no consideration of timing deadlines.
- Real-time Operating System (RTOS):
  - Still relatively easy to program.
  - Not faster; but provides decent bounding on jitter.
- Field Programmable Gate Array (FPGA):
  - Not simple to program.
  - Perfectly timing deterministic.

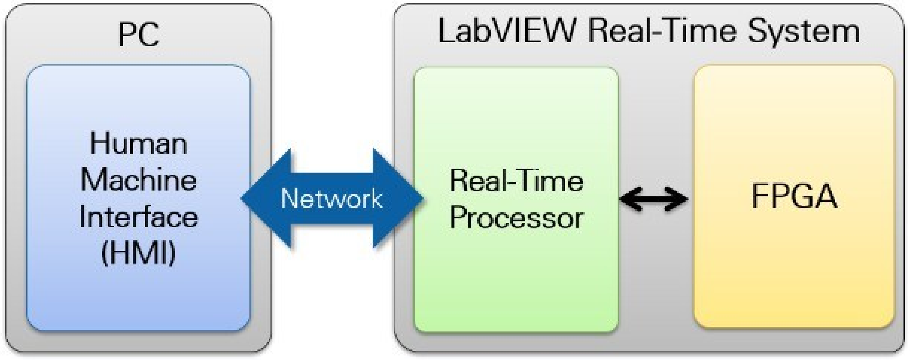


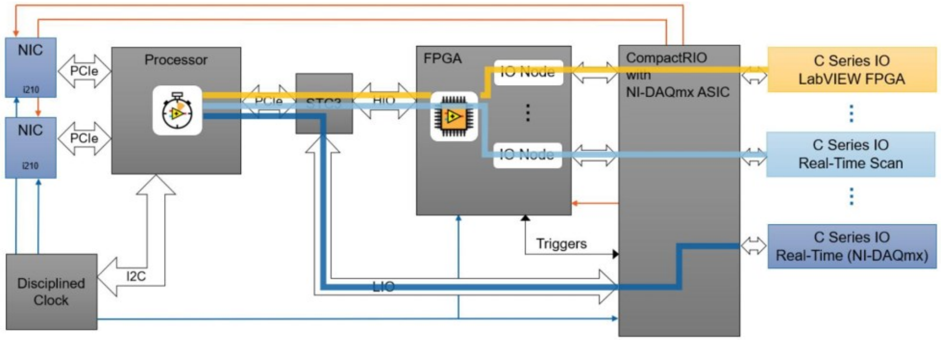


Consequences of Missing Information at Strict Latencies

The exchange of data across processing systems has several key challenges for challenges:

- To use updated models, there needs to be some sort of AI or decision-maker one level up.
- To maintain high-rate transfers (to and from an FPGA), an RTOS will be needed between an GPOS and an FPGA.
- Information for decision making is typically done though a “publish and subscribe” transfer protocol, but this will introduce large uncertainties in timing.
- FPGAs are well suited for doing repetitive tasks, and as such should be considered for data fusion and filtration of sensor signals.





Embedded System Components. https://www.ni.com/en-us/support/documentation/supplemental/16/understanding-communication-options-between-the-windows-hmi--rt-.html

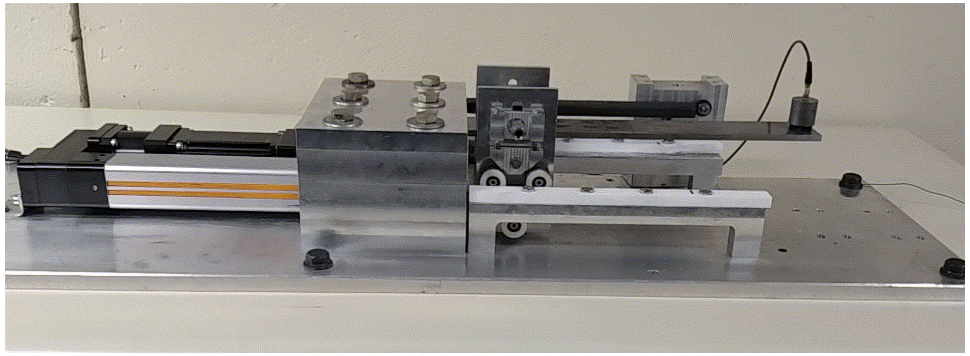
Operating System/Hardware Selection

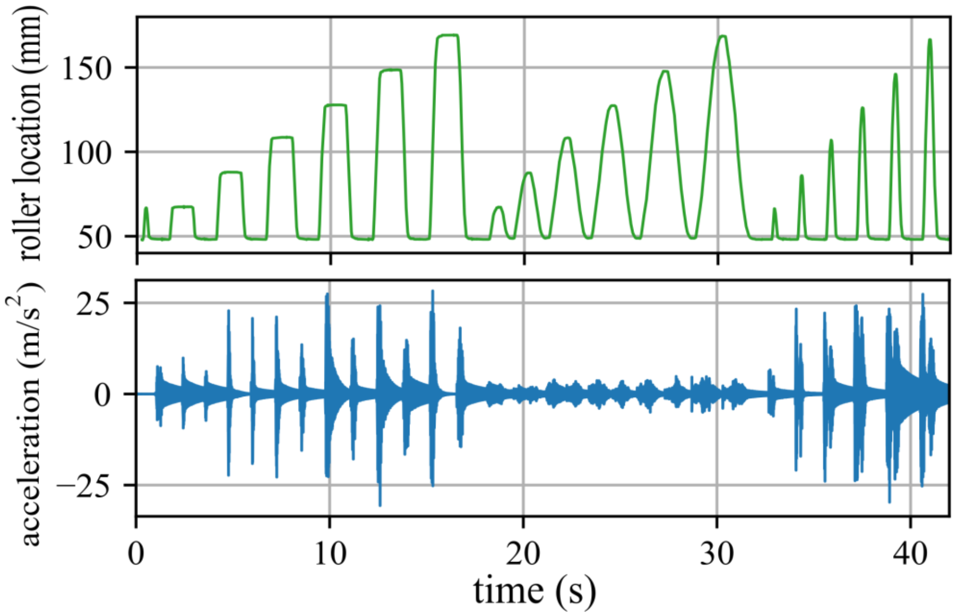
UofSC South Carolina

Experimental System used for this work

DROPBEAR experimental testbed:

- The Dynamic Reproduction of Projectiles in Ballistic Environments for Advanced Research (DROPBEAR) was used to generate the experimental data in this work.
- Cantilever beam with a controllable roller to alter the state.
- Acceleration and pin location are recorded.
- Dataset available on GitHub at: https://github.com/High-Rate-SHM-Working-Group/Dataset-2-DROPBEAR-Acceleration-vs-Roller-Displacement

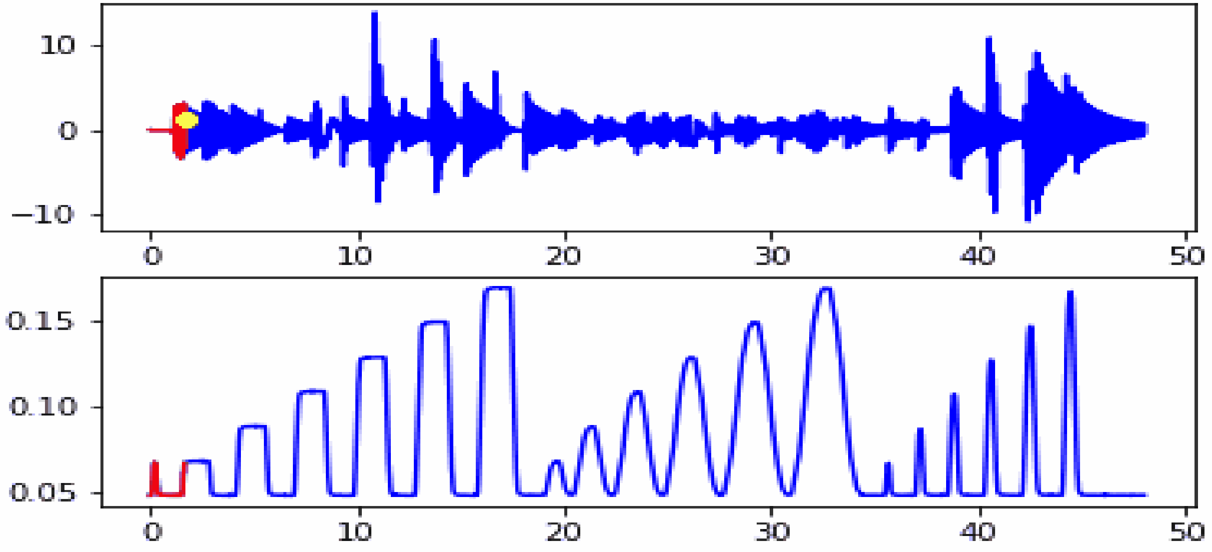


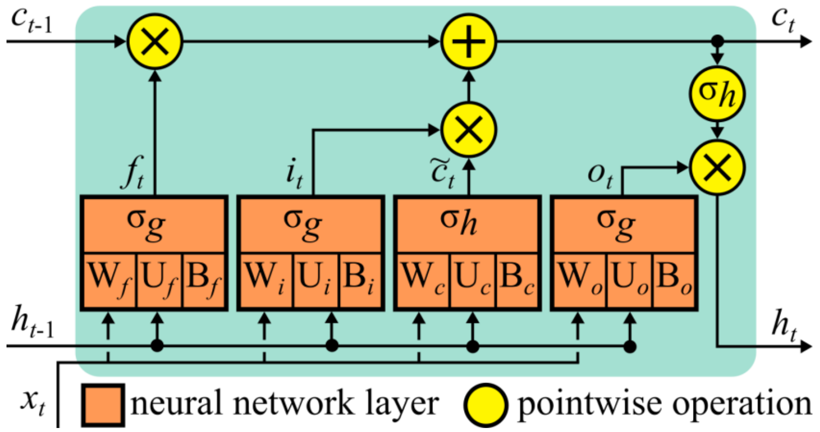


LSTM-based Real-time State Estimation

In this work:

- Long short-term memory (LSTM) models are used for real-time state estimation.
- These data-driven models are trained offline on pre-recorded data.

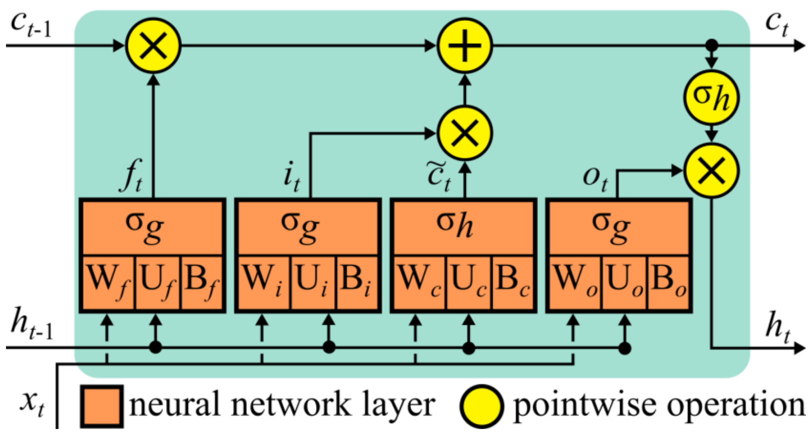


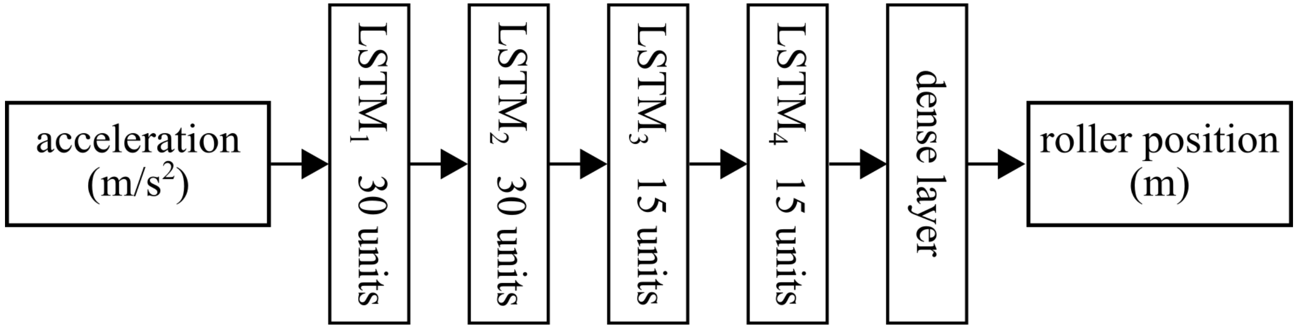


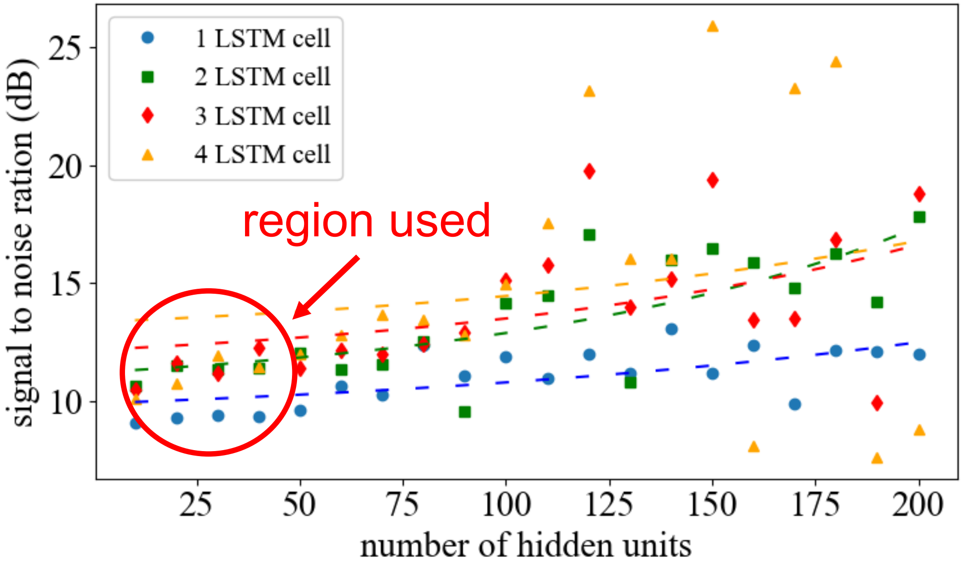
Long Short-term Memory Model

LSTM features and development:

- LSTMs are a Recurrent Neural Network (RNN) that propagates through long- and short-term memory forms.
- Four stacked LSTM cells (30, 30, 15, 15 units) with a fully connected layer at the output.







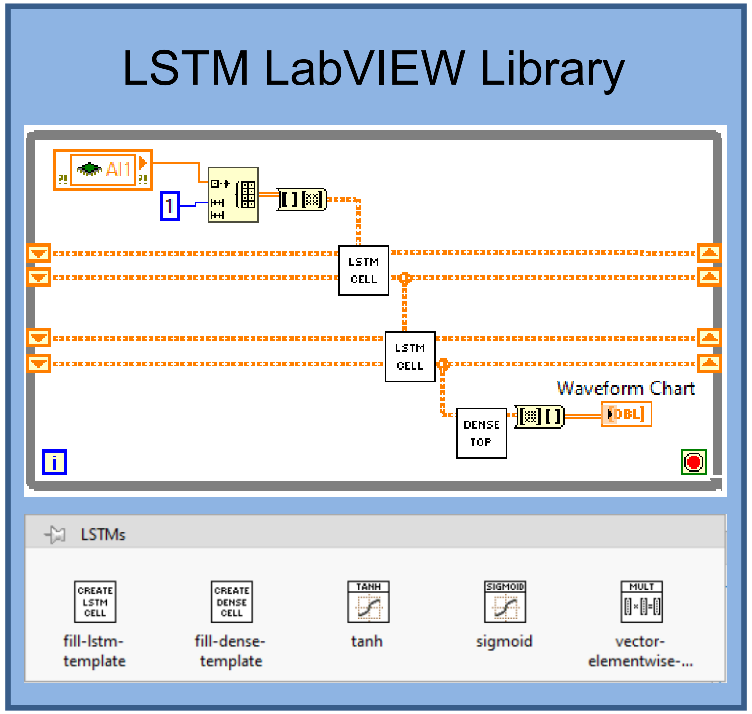
Model Deployment on a Real-Time Operating System (RTOS)

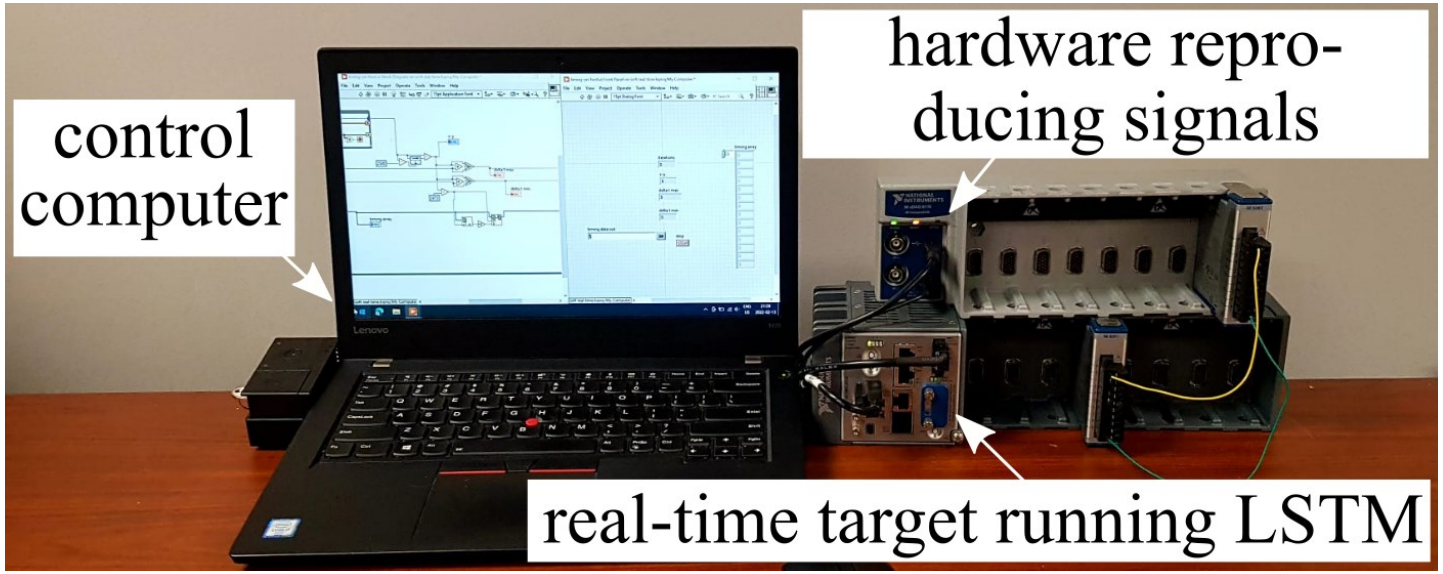
South Carolina

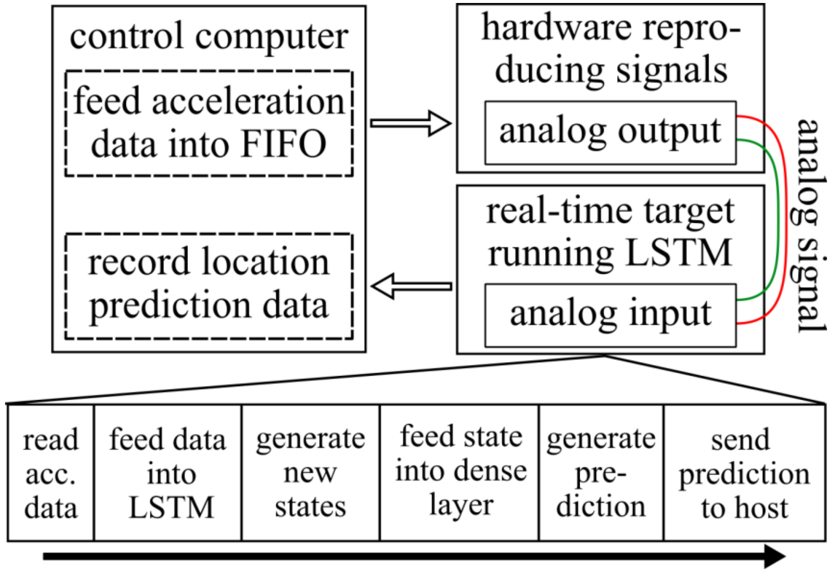
Model Deployment on RTOS

Real-time validation performed on an embedded system running:

- The experimental setup consisted of two subsystems:
  - **Hardware reproducing Signals** reproduces the dataset.
  - **Real-time Target re-**digitizes and feeds the input into the LSTM.
- Data is sampled at 1250 S/s; a prediction is made every 800 μs.
- State predictions are returned via a FIFO buffer to PC.



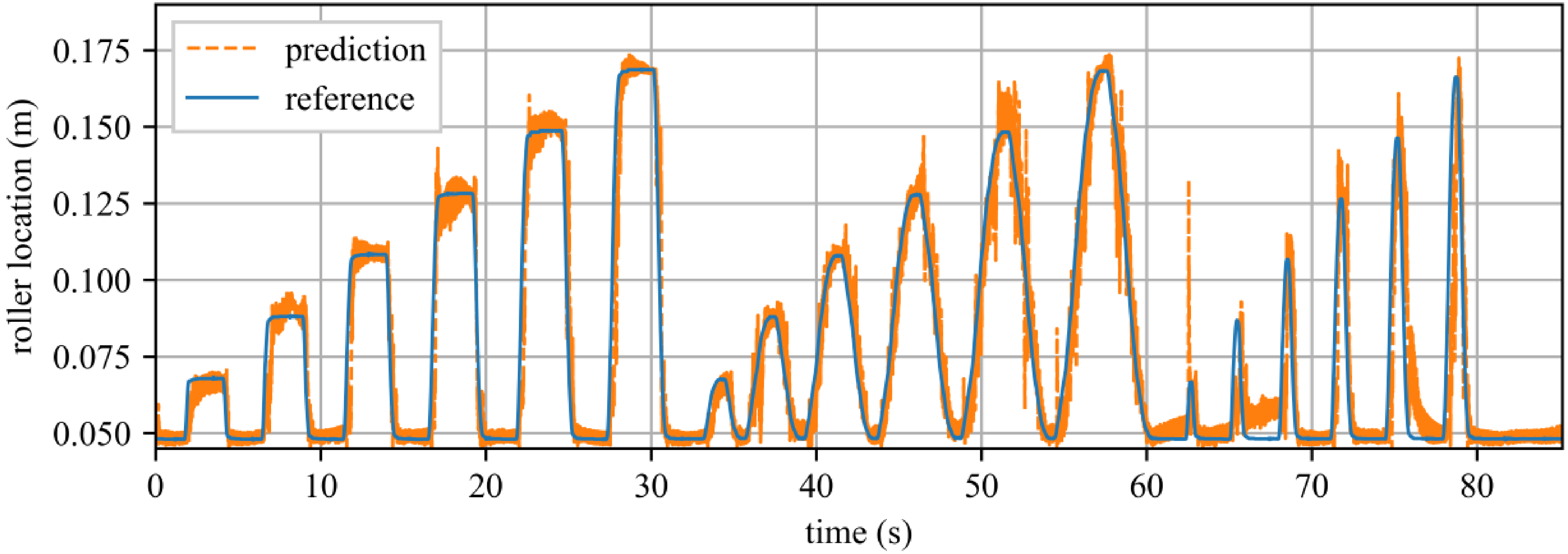




Real-time LSTM Modeling Results

LSTM model performance results:

- SNR_{dB} 17.4888 dB.
- RMSE of 11.471 m mm.
- LSTM traces reference pin location closely.



Real-time LSTM Modeling Results

LSTM model timing results:

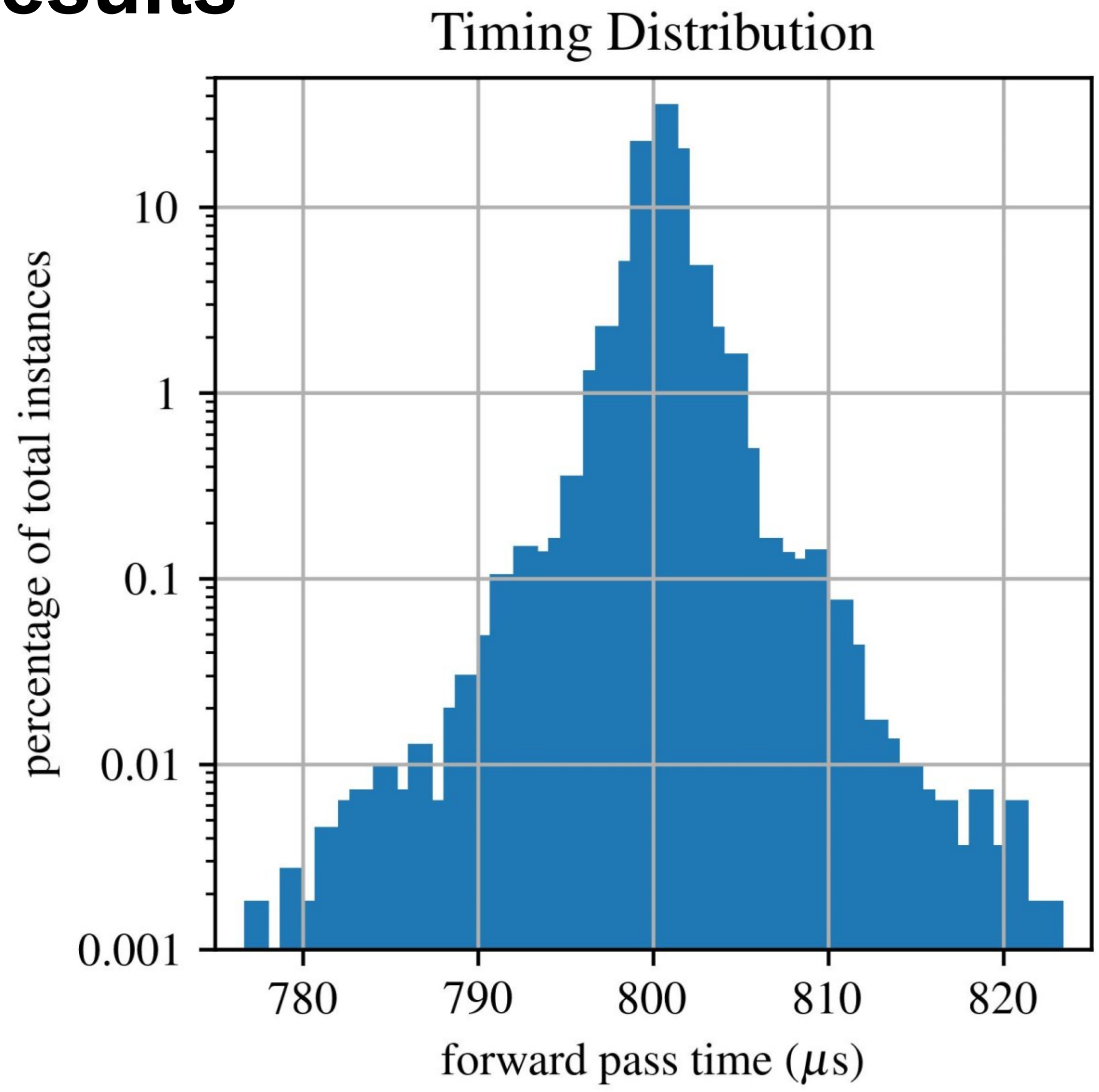
- Average: 800 μs.
- Standard deviation: 1.79 μs.
- Max overshoot: 26 μs.

Timing accuracy results:

- Execution-time jitter as expected.
- Timing follows a normal distribution.

Intel Atom® Processor E3825

- Total Cores: 2 (2 threads)
- Processor Base Frequency: 1.33 GHz
- Cache: 1 MB L2 Cache
- Use Conditions: Automotive, Embedded

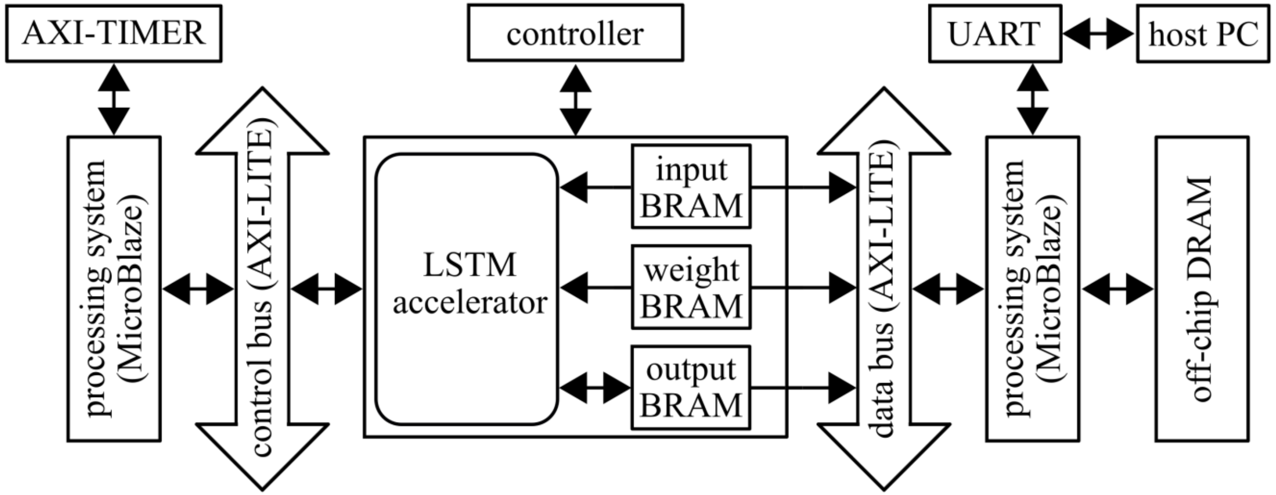


Model Deployment on a Field Programable Gate Array (FPGA)

Model Deployment on FPGA

LSTM model deployed on a Xilinx Virtex 7 (VC707) FPGA:

- Implemented in both 16-bit fixed point.
- Developed an LSTM hardware accelerator where data in and out the FPGA is pre and post-processed with the MicroBlaze soft core processor.

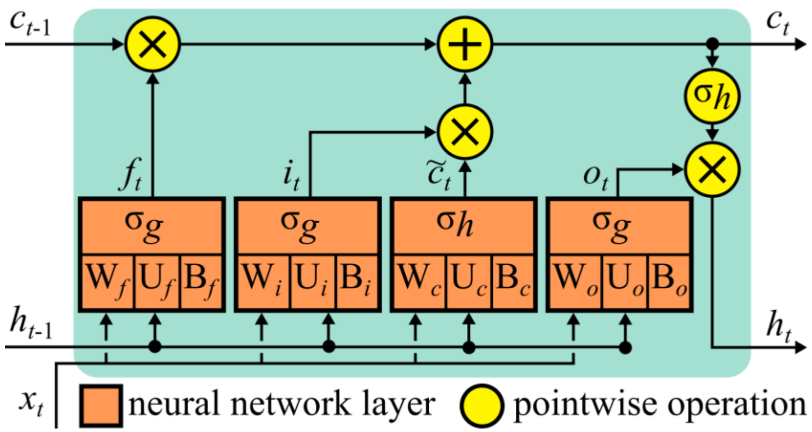


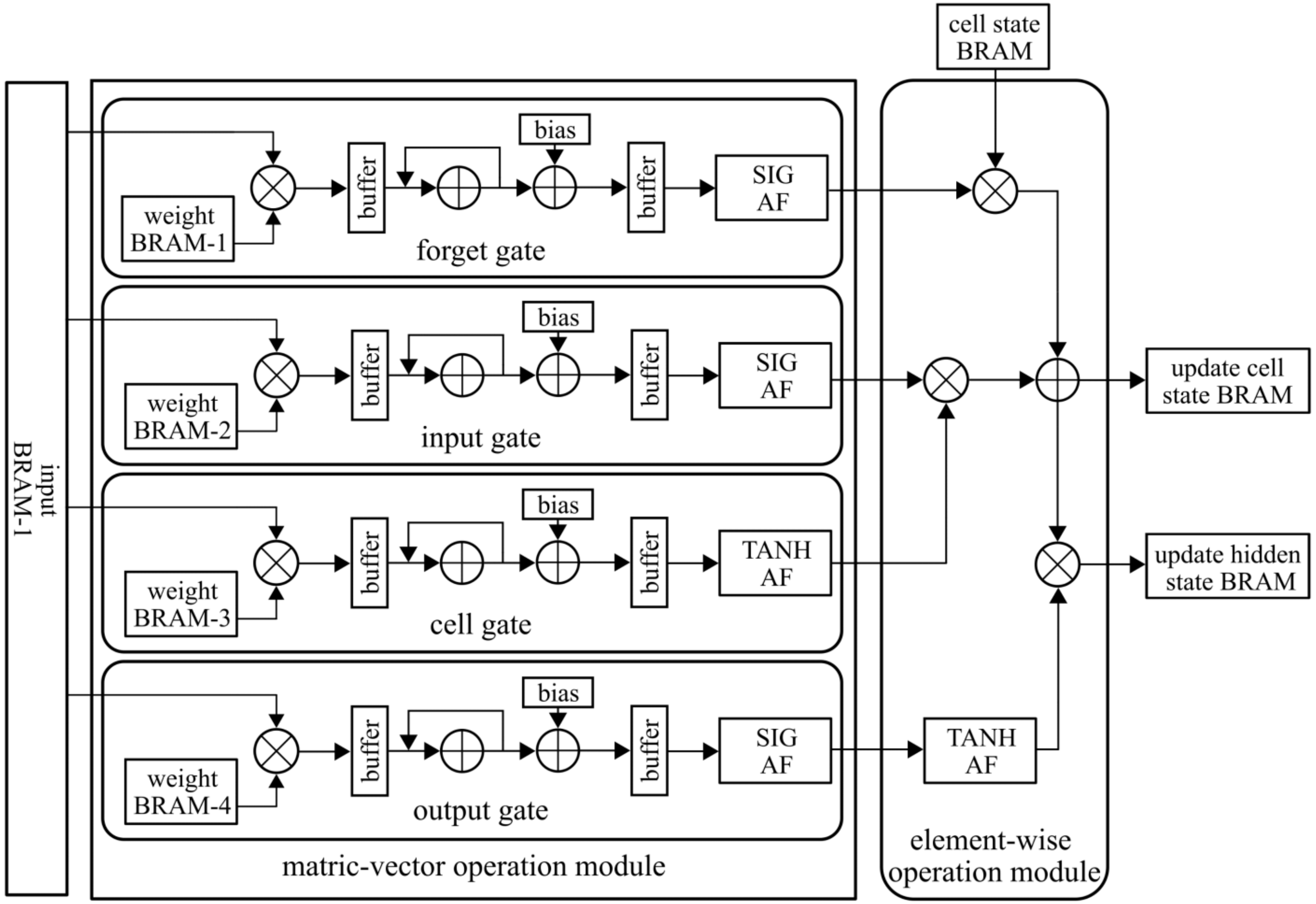
Xilinx Virtex 7 (VC707)

South Carolina

LSTM deployment on an FPGA

The developed hardware accelerator is split up into the LSTM's gates for deployment.

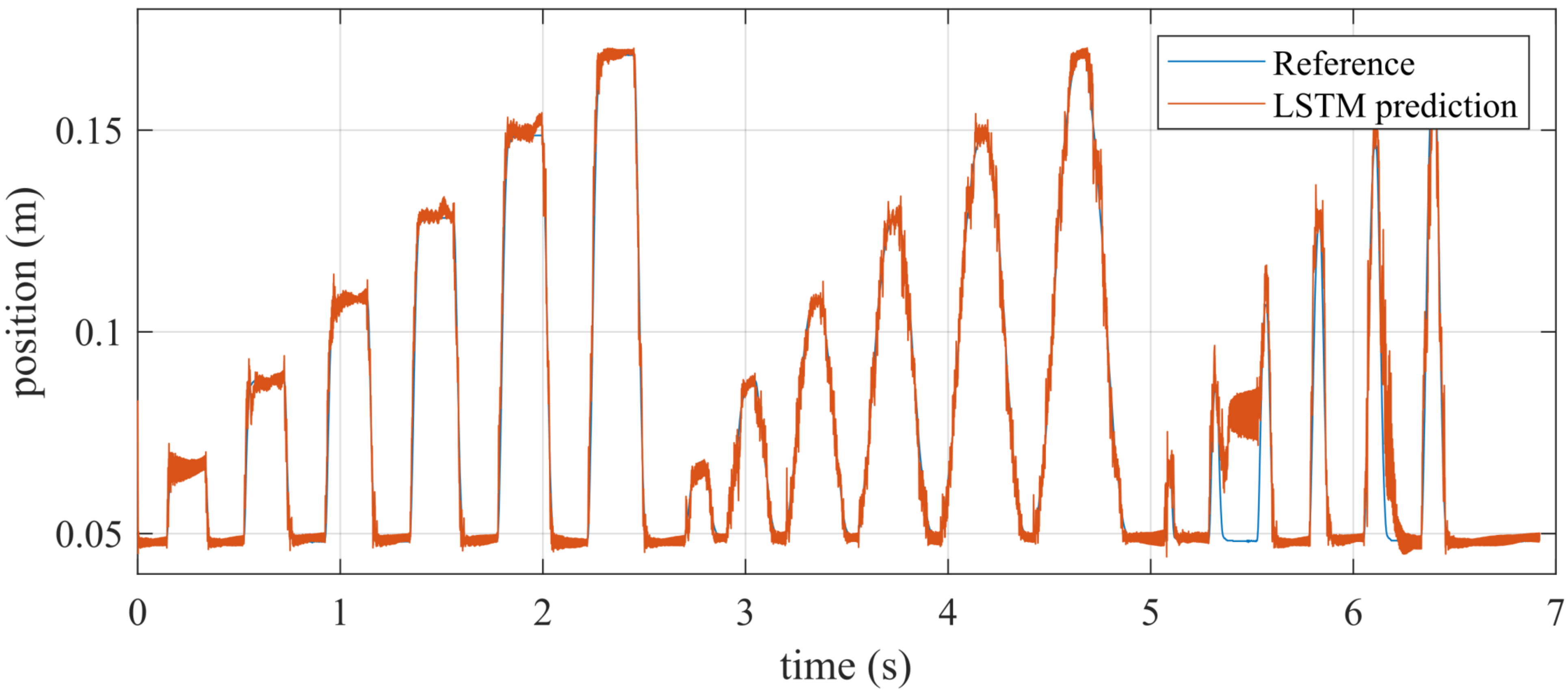




Real-time LSTM Modeling Results

16-bit fixed point model performance:

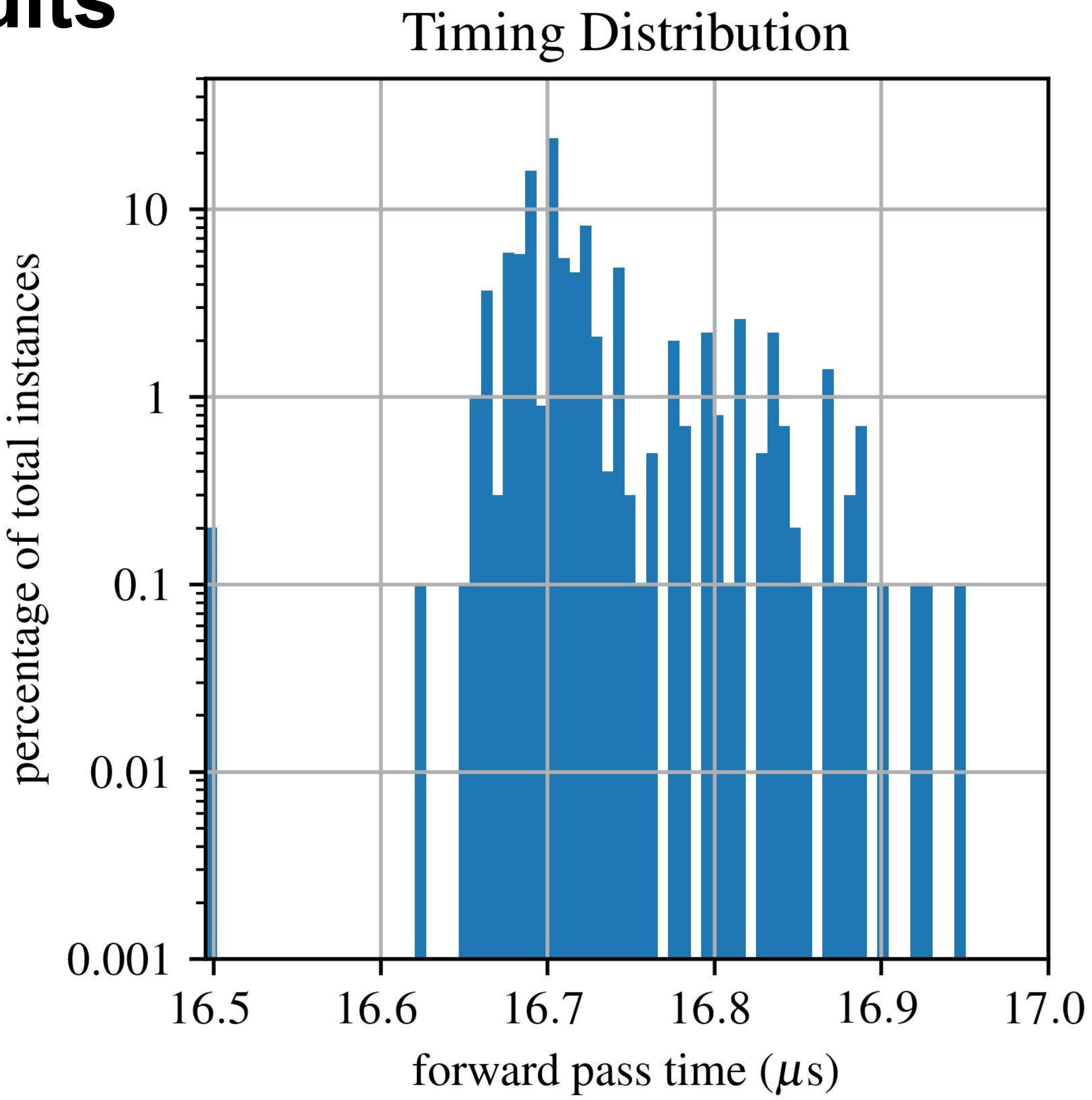
- SNR_{dB} of 19.54 dB.
- RMSE of 9.1 mm.



Real-time LSTM Timing Results

16-bit fixed point model performance:

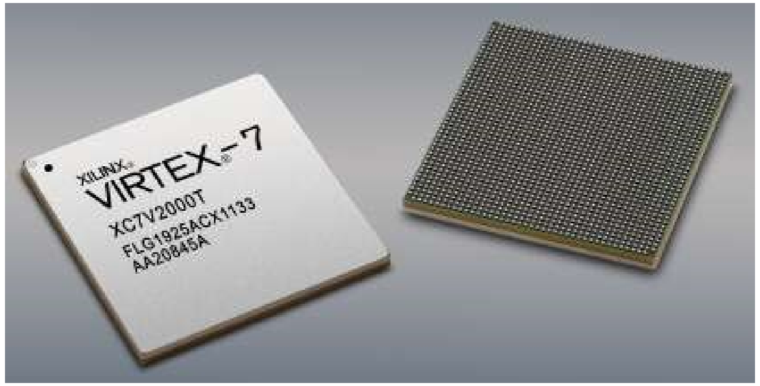
- Time step of 16.7 μs.
- Standard deviation: 0.0509 μs.
- 50X speed up over RTOS.



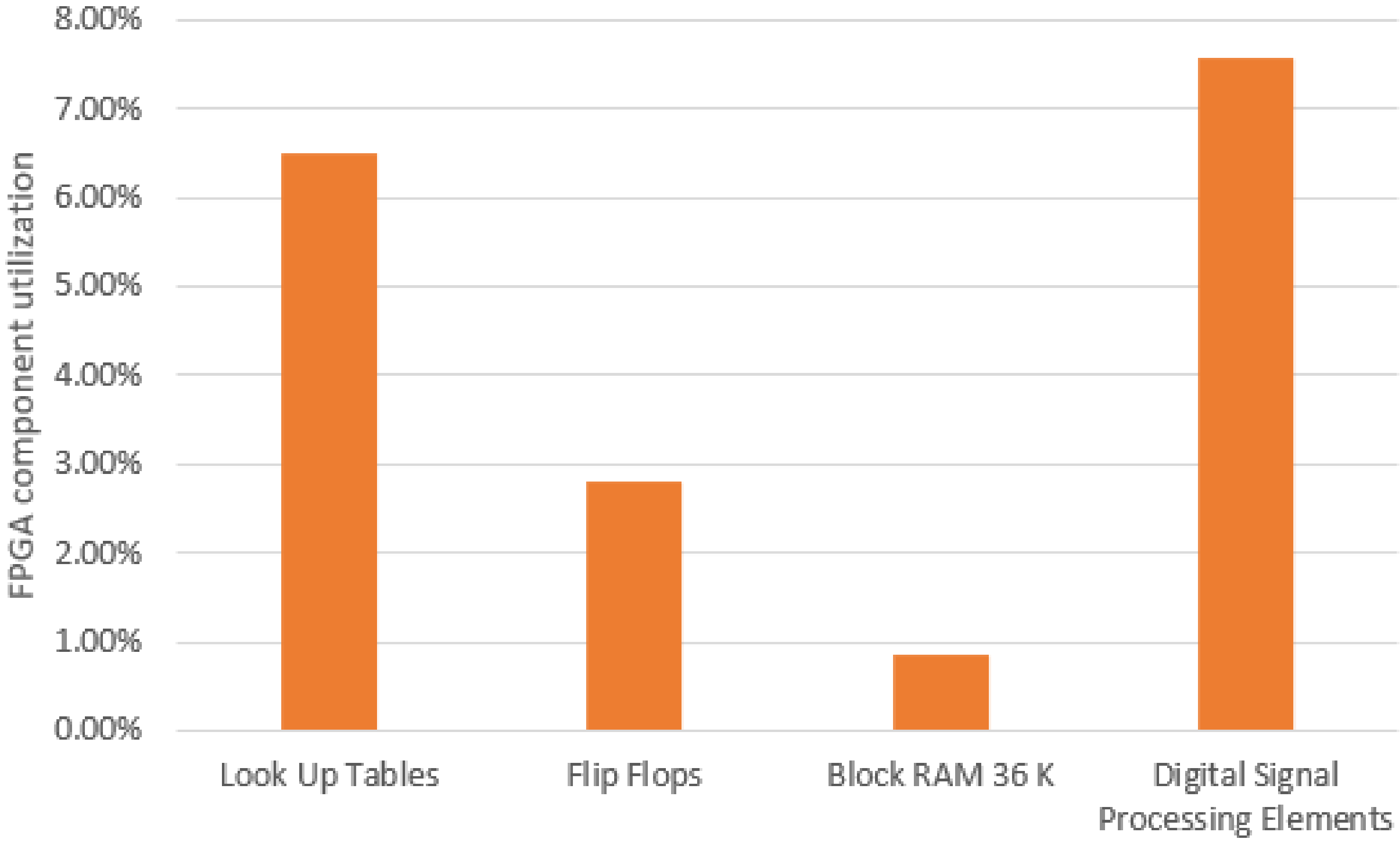
FPGA Resource Utilization Results

Synthesized for the Xilinx Virtex 7 (VC707):

- Consume less the 10% of FPGA resources.
- Has potential for deployment to much smaller FPGAs



https://www.eetimes.com/new-xilinx-virtex-7-2000t-fpga-provides-equivalent-of-20-million-asic-gates/

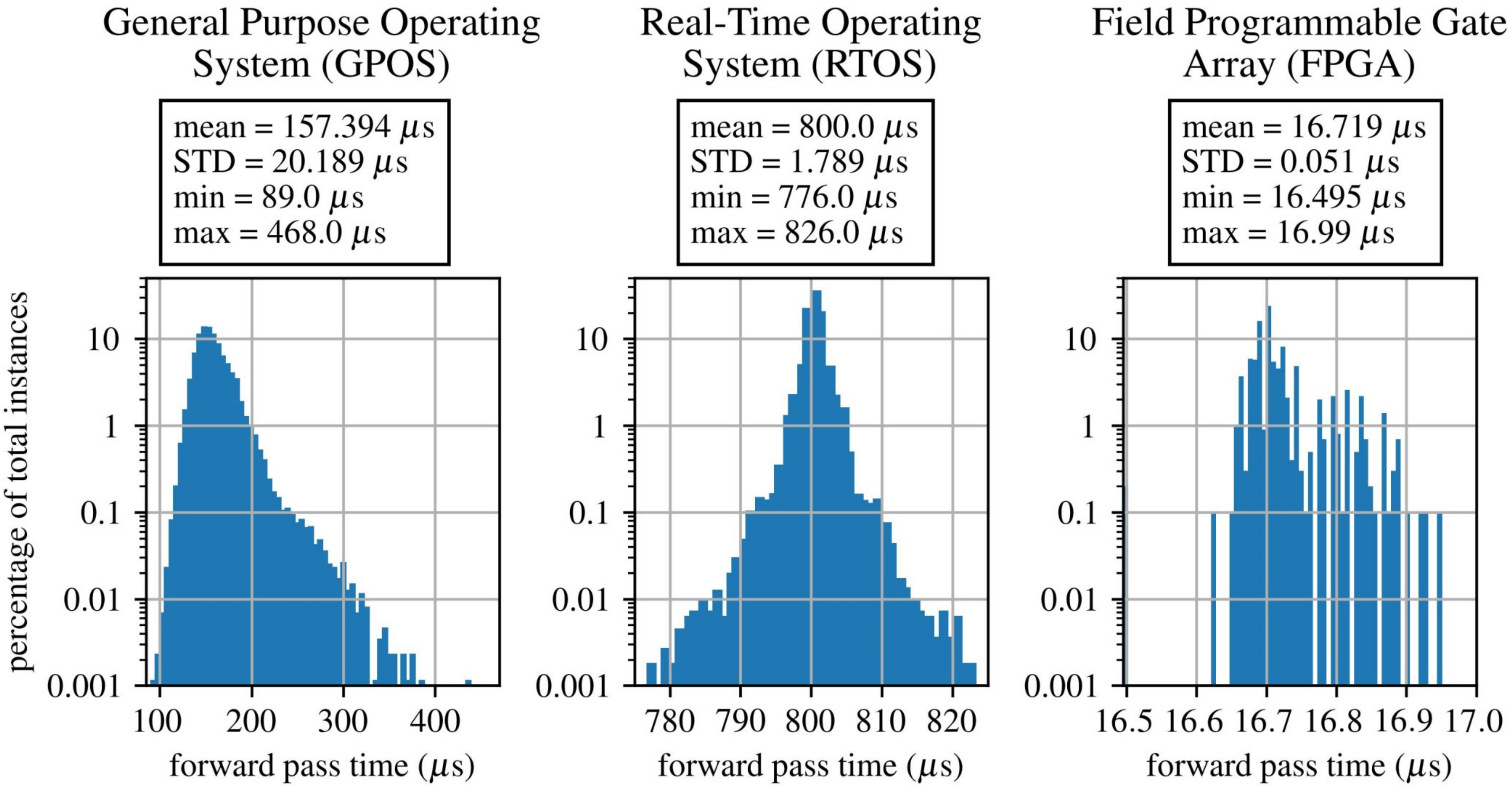


Look Up Tables	Flip Flops	Block RAM 36 K	Digital Signal Processing Elements
126633	109186	229	212

GPOS vs RTOS vs FPGA

South Carolina

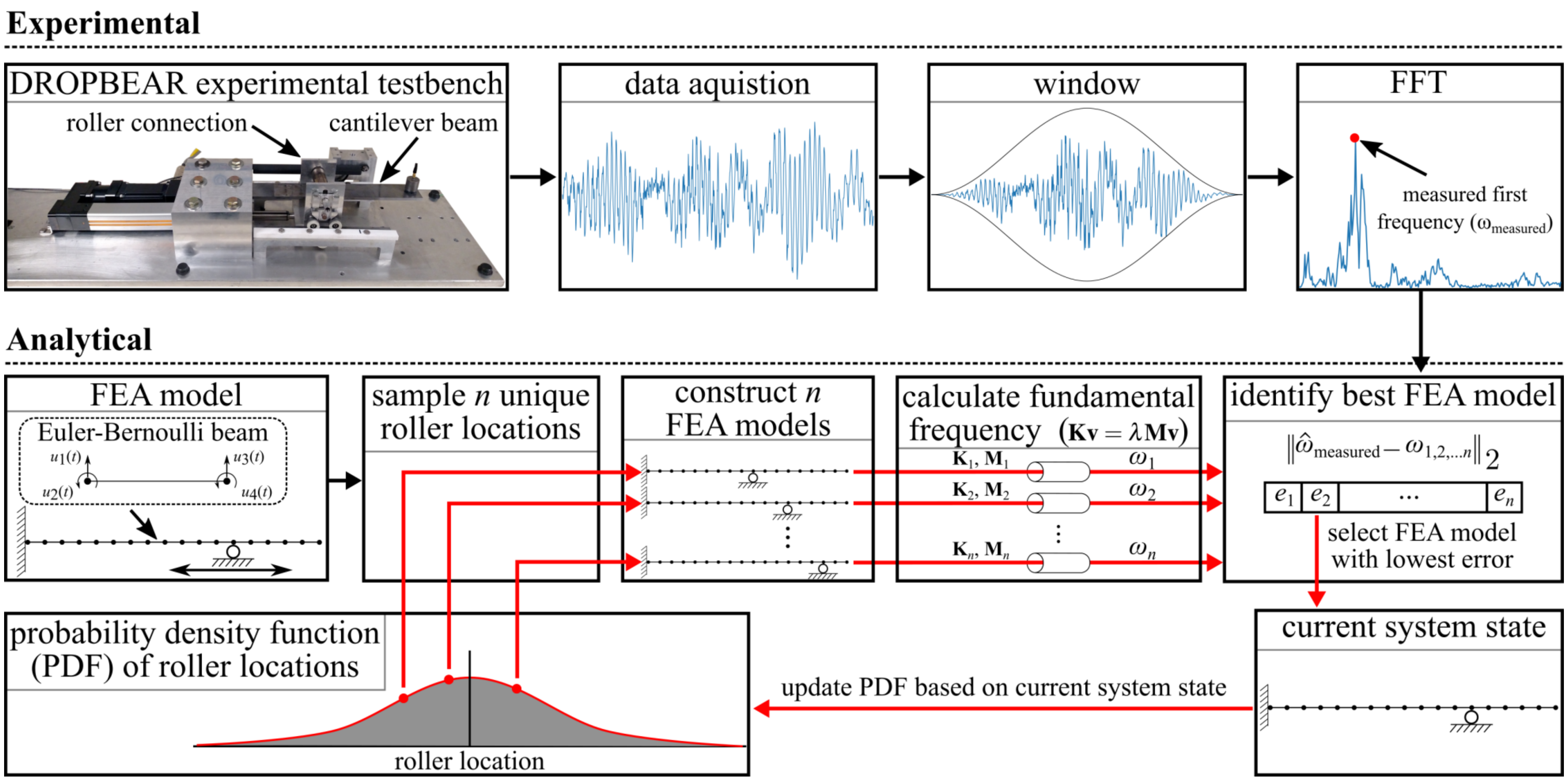
Timing characteristics by hardware implementation



Example: Memory Bandwidth Limitations

Real-Time Model Updating Through Error Minimization

A frequency-based model updating technique was developed to update an FEA model of the system.



Downey A., et al,. “Millisecond Model Updating for Structures Experiencing Unmodeled High-Rate Dynamic Events” *Mechanical Systems and Signal Processing* ***138***, 2020

Eigen Value Problem - Can we just solve it faster?

- General Eigenvalue solutions are a well studied problem.
- Hardware accelerators do exist, but their throughput is limited by communication bandwidth.

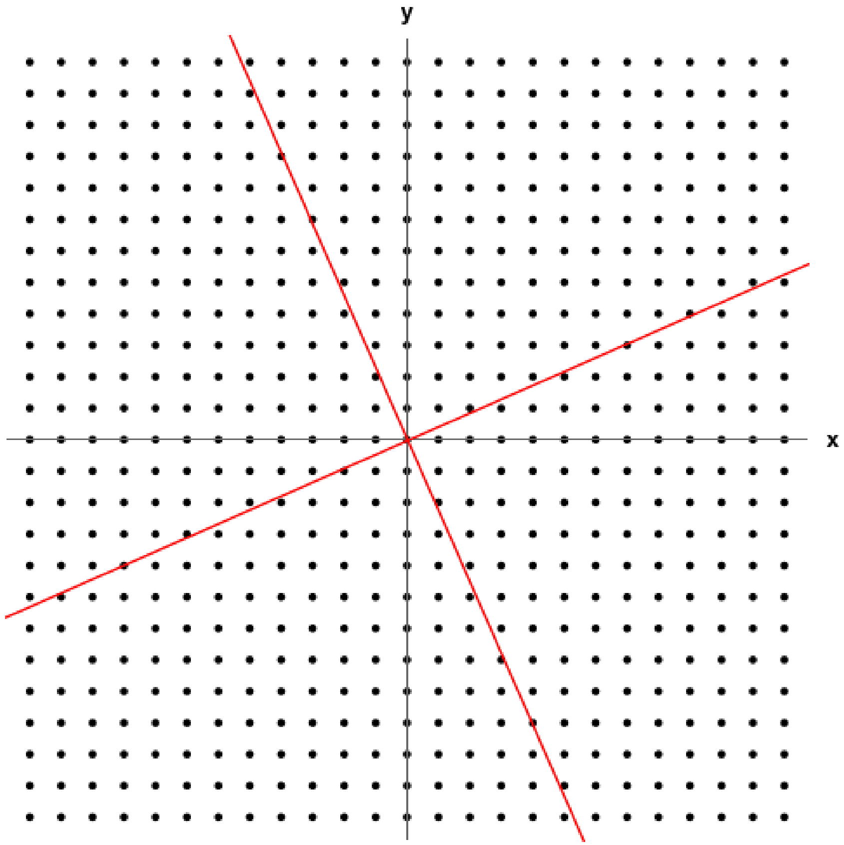
$$\mathbf{M}_1\ddot{x} + \mathbf{K}_1 x = 0$$

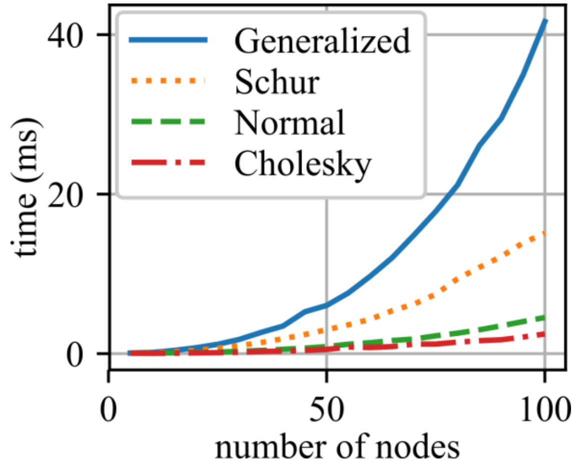
$$\det[\mathbf{K}_1 - \boldsymbol{\lambda}\mathbf{M}_1] = 0$$

$$[\mathbf{K}_1 - \boldsymbol{\lambda}\mathbf{M}_1]\mathbf{U}_1 = 0$$

$$\boldsymbol{\lambda} = \begin{bmatrix} \omega_1^2 & 0 & 0 & 0 \\ 0 & \omega_2^2 & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & \omega_n^2 \end{bmatrix}$$

$$\mathbf{U}_1 = [\vec{u}_1^1 \quad \vec{u}_2^1 \quad \cdots \quad \vec{u}_n^1]$$



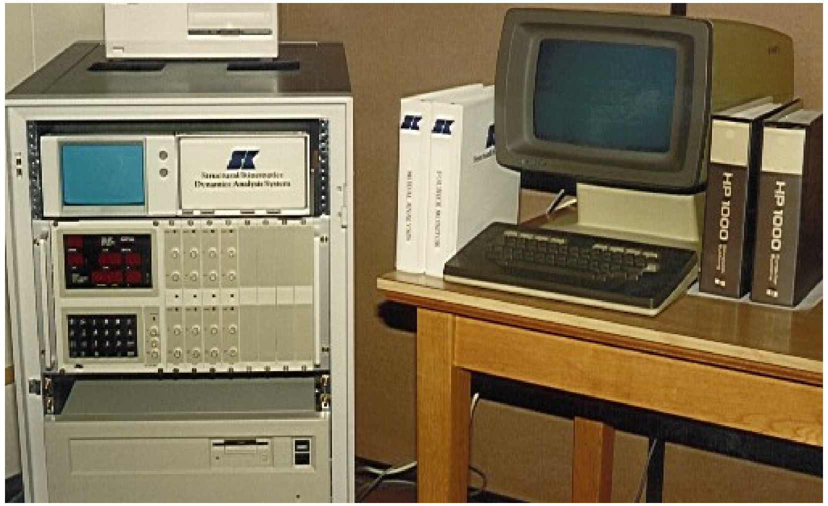


Developed a specialized Cholesky-Jacobi method formulated specifically for this challenge; a 66 node FEA model can be solved for within the 1 ms.

LEMP usage in the ‘70s and ’80s

LEMP enabled these calculations to be done very efficiently on very slow desktop computers.

- Structural Measurements Systems (SMS) sold a custom hardware and software setup.
- This was before the “personal computer” stage.



HP1000/A700 w/DIFA modal analysis system

SMS modal software called SDM used LEMP

HP3000 desktop running “Rocky Mountain BASIC”

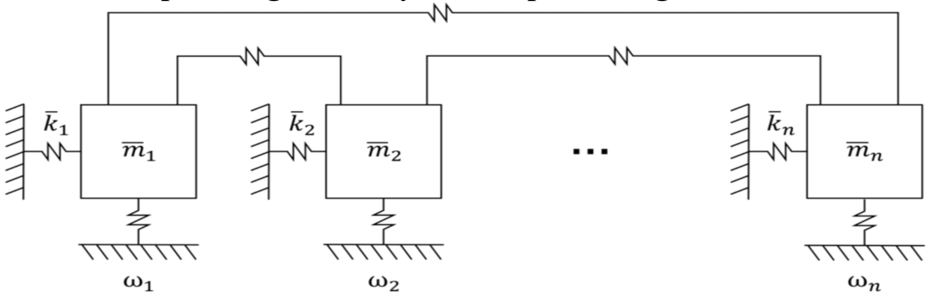
HP5423 first dedicated FFT/Modal system - 1979

All images and knowledge curtesy of Peter Avitabile Professor Emeritus, Co-Director - Structural Dynamics & Acoustic Systems Laboratory at the University of Massachusetts Lowell

Local Eigenvalue Modification Procedure (LEMP)

n independent single DOF systems representing the initial state

Coupled single DOF systems representing the altered state



	Initial State	Modification →	Altered State	
Physical Space	$[\mathbf{M}_1], [\mathbf{K}_1]$ →	$[\Delta\mathbf{M}_{12}], [\Delta\mathbf{K}_{12}]$ →	$[\mathbf{M}_2], [\mathbf{K}_2]$	‘*n*’ Physical DOF
Modal Transformation	$\{x\} = [U_1]\{p_1\}$	$$\frac{-1}{\alpha} = \sum_{r=1}^{m} \frac{v_r^2}{\omega_r^2 - \Omega_r^2}$$ **Solved using Divide and Conquer method**	$\{x\} = [U_2]\{p_2\}$	*m*<<*n*
Modal Space	$[\omega_1^2], [U_1]$ →	$\{p_1\} = [U_{12}]\{p_2\}$ →	$[\Omega_2^2], [U_2]$	‘*m*’ Modal DOF

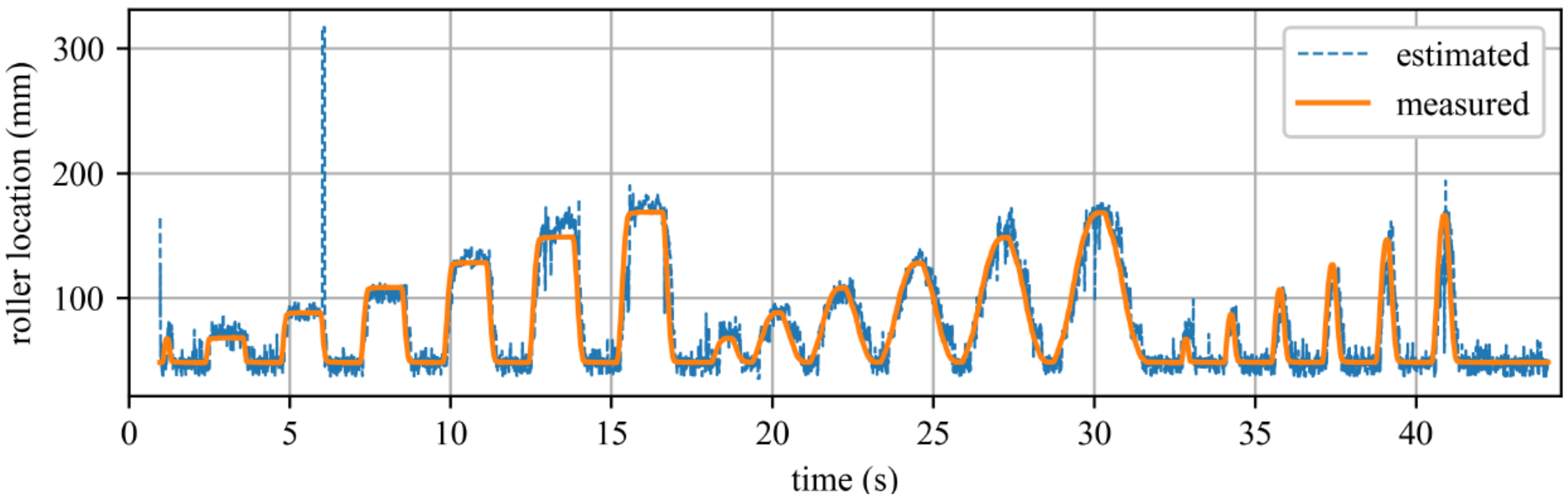
Avitabile, P., “*Twenty Years of Structural Dynamic Modification- A Review*,” *Sound and Vibration*, pp. 14-25. 2003
Drnek, C. R., “*Local eigenvalue modification procedure for real-time model updating of structures experiencing high-rate dynamic events*,” (2020).

LEMP State Estimation Results

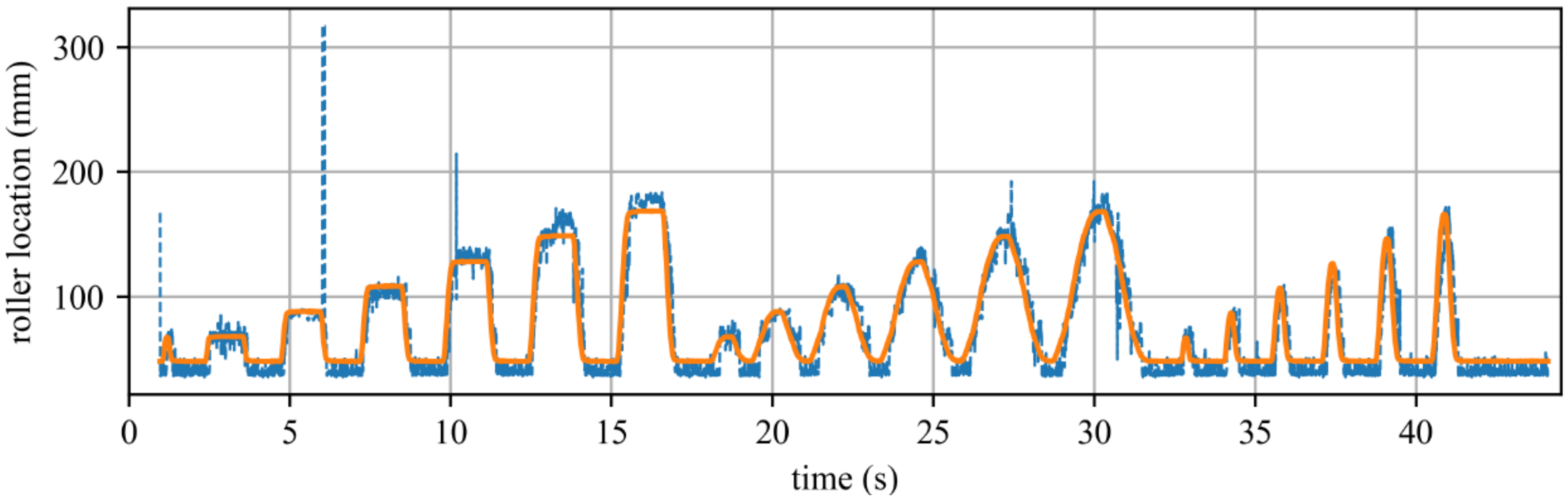
LEMP has been shown to:

- Have similar accuracy to the generalized eigenvalue solver
- Be robust to sensor noise
- Be capable of working with various error estimator developed for the project

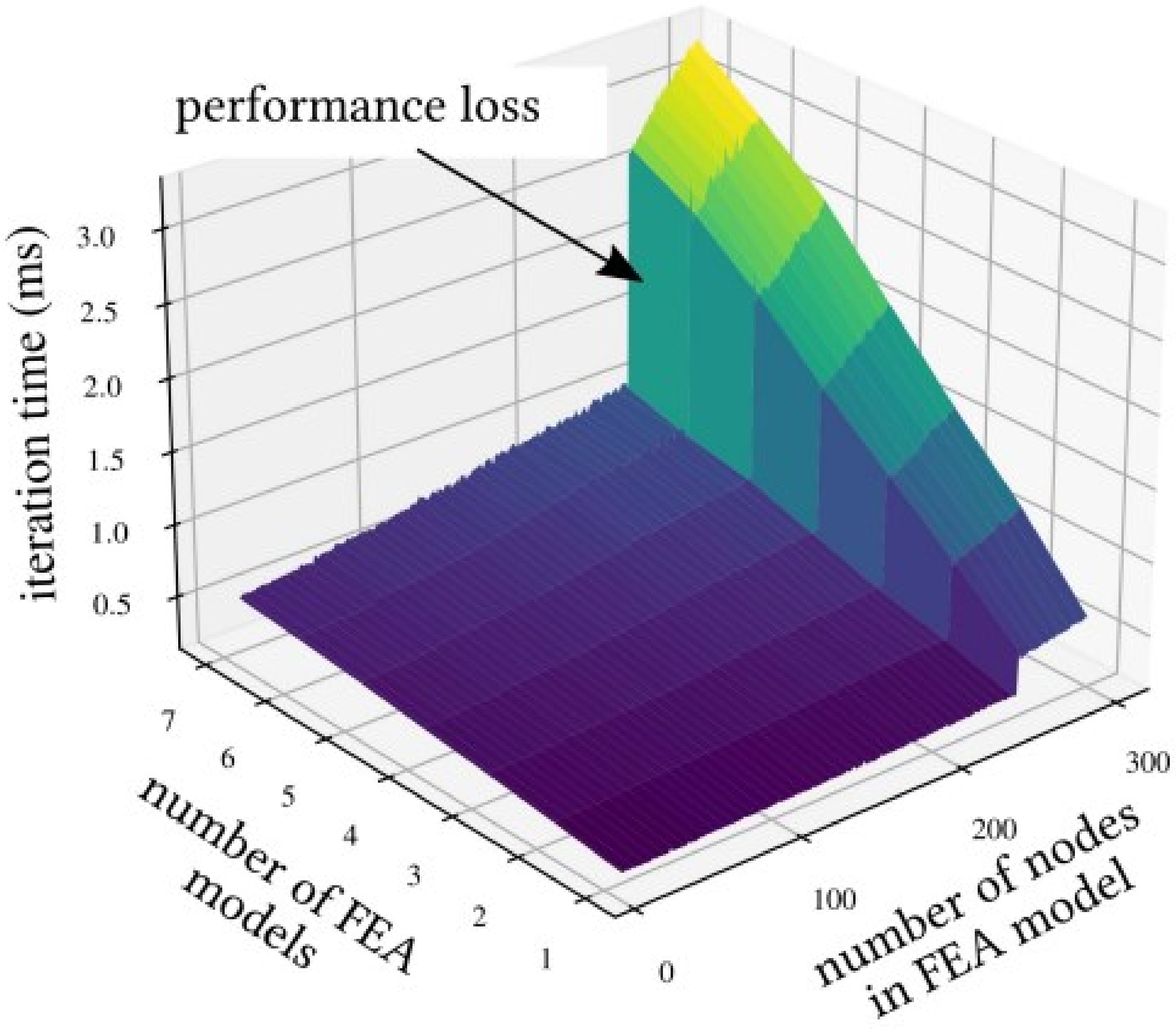
generalized eigenvalue solver



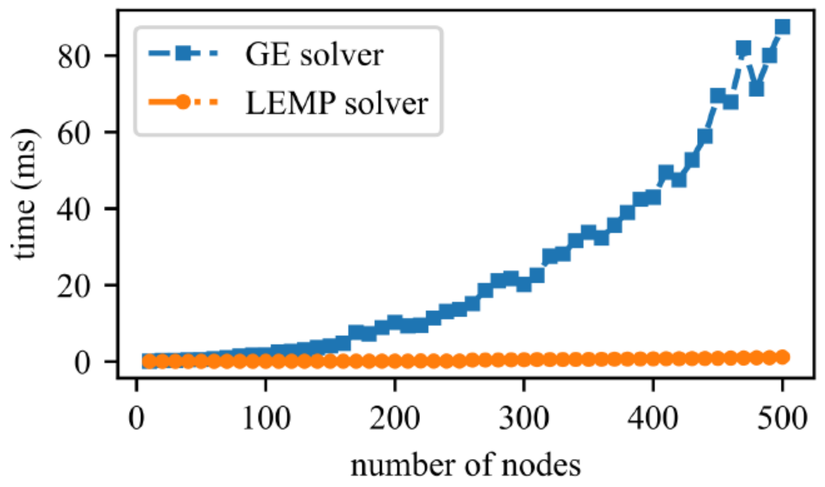
LEMP solver



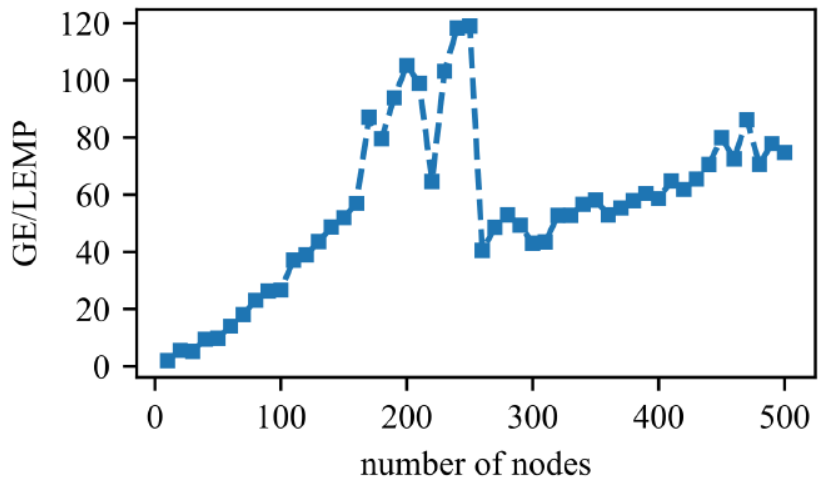
Memory challenges – Hardware/software Co-design



Algorithm Timing



Speed up Factor



DISCUSSION

Open-Source Codes and Data Sets

- LEMP solver: https://github.com/ARTS-Laboratory/Paper-Real-time-Structural-Model-Updating-using-Local-Eigenvalue-Modification-Procedure
- Secular equation solver: https://github.com/ARTS-Laboratory/Paper-Development-of-a-Real-time-solver-for-the-Local-Eigenvalue-Modification-Procedure
- DROPBEAR dataset: https://github.com/High-Rate-SHM-Working-Group/Dataset-2-DROPBEAR-Acceleration-vs-Roller-Displacement
- Open-Source library for Deploying LSTMs to the NI Linux Real-time Operating System at: https://github.com/ARTS-Laboratory/LabVIEW-LSTM

Contact Information: Austin Downey
Email: austindowney@sc.edu
Github: https://github.com/austindowney
Github-Lab: https://github.com/Arts-laboratory/

South Carolina