EM-Flow: Advanced Electromagnetic Control Flow Verification for Embedded Systems

Carson Stillman[1], Jennifer Sheldon[1], Ian Y. Garrett[2], Patrick Traynor[1], Ryan M. Gerdes[2], Sara Rampazzi[1,] and Kevin R. B. Butler[1]

[1]University of Florida, [2]Virginia Tech



Low-Power Embedded Devices

- Low-power embedded devices form a critical backbone of the modern world
  - They find broad use in agriculture, IoT, and medical devices
- Yet they are difficult to secure
  - Direct memory access and no randomization makes attacks simple







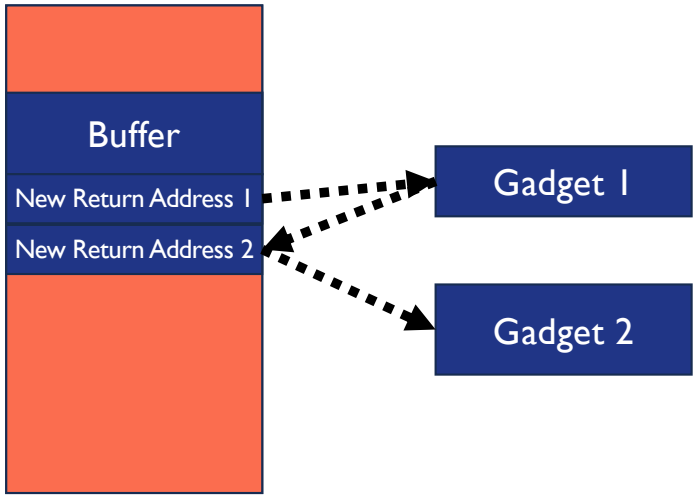
Control Flow Attacks

- Control flow attacks function by deviating the program's control flow to accomplish some malicious objective
- Some examples include buffer overflow attacks, and return-oriented programming attacks

Buffer Overflow Attack

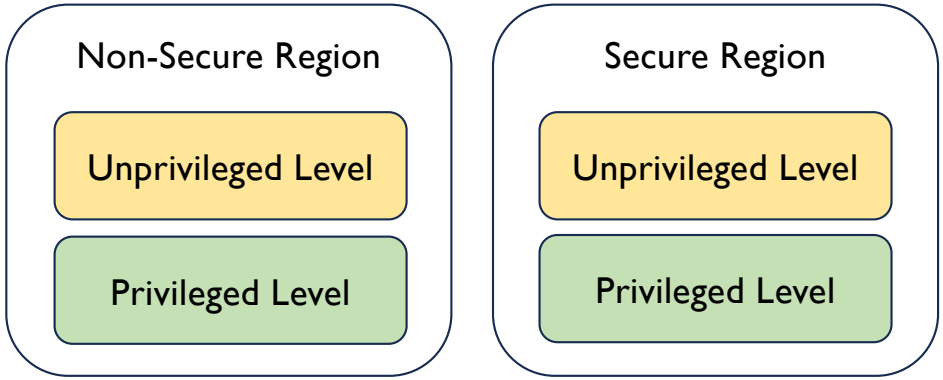


ROP Attack



Standard Solutions

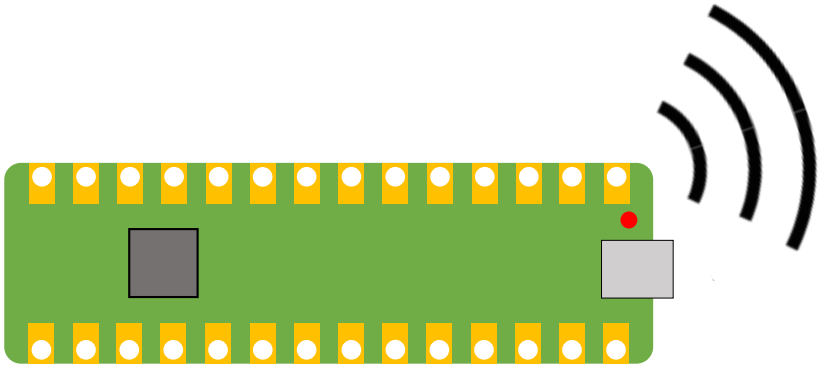
- Trusted execution environments are expensive and unavailable on all commodity hardware, such as the ARM Cortex-M0
- Memory Protection Units, are limited by a small number of configurable regions



Using Device Physics

- We propose using the Electromagnetic (EM) emissions emitted as a byproduct of computation to verify the execution of software on embedded devices

- In this work we evaluate our framework on two devices, the Arduino Uno and the Raspberry Pi Pico, and six target programs



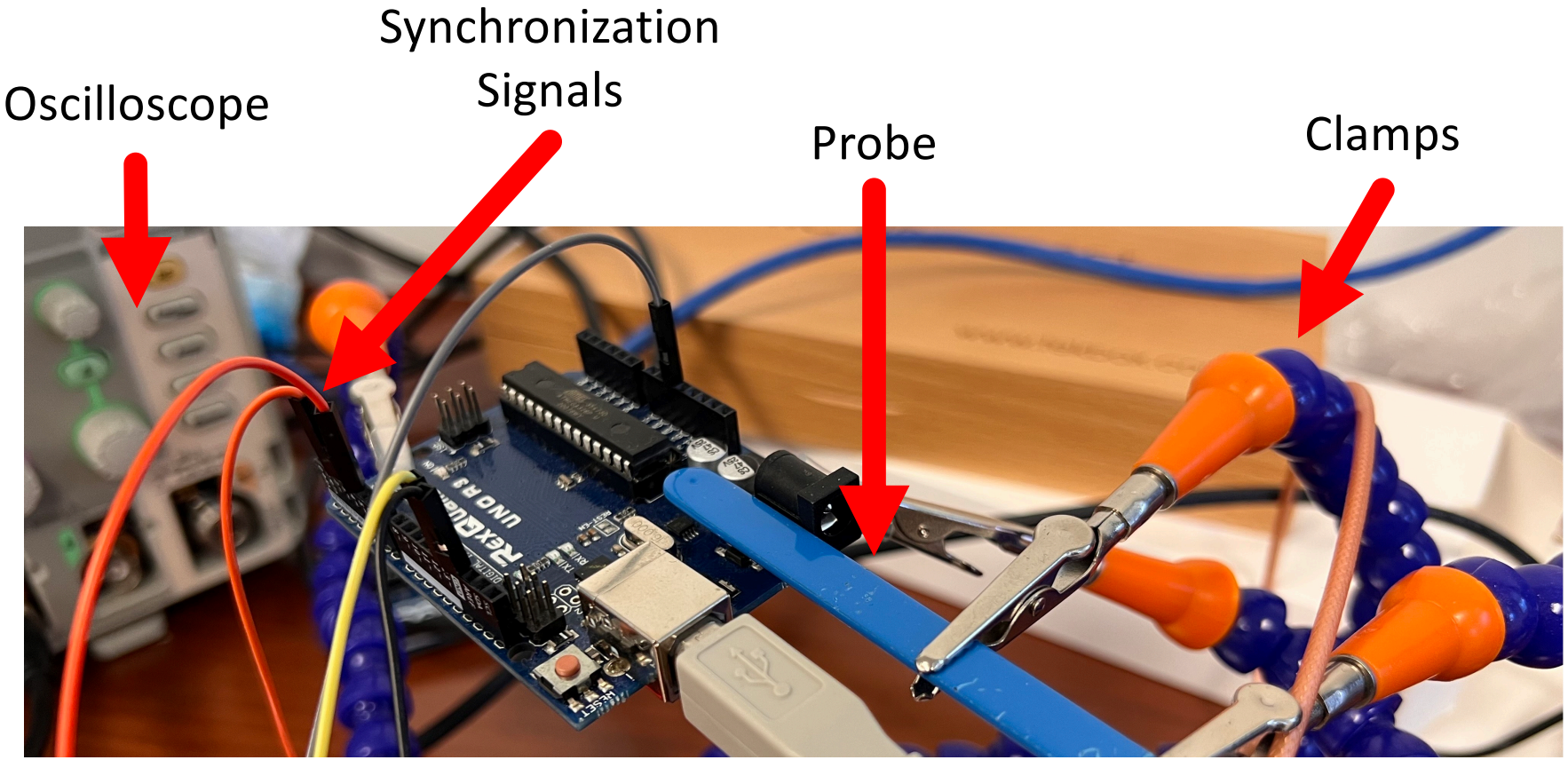
Research Questions

RQ1 As practical control flow attacks launched on embedded devices require at least three clock cycles to complete, are we able to identify these fine-grained control flow deviations in our target case studies?

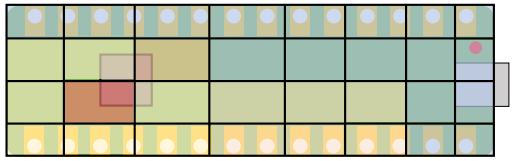
RQ2 Does our framework detect subtle control flow deviations to verify correct execution of the security-critical sections of the target program?

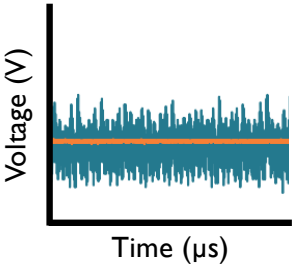
RQ3 Are the features we identify to classify basic blocks robust and transferable to new devices?

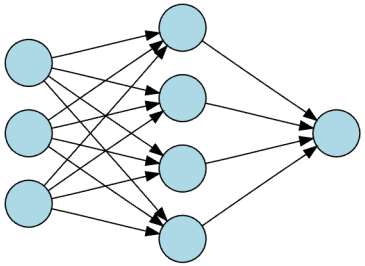
Experimental Setup



EM Verification Pipeline



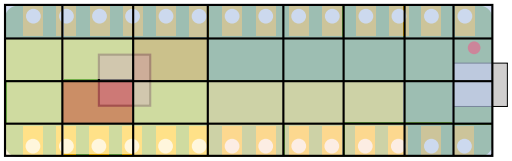


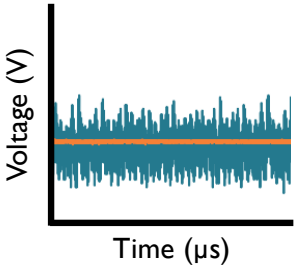


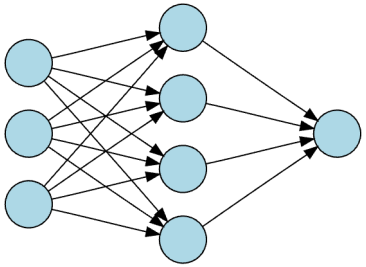


EM Leakage Detection → Basic Block Collection → Basic Block Classification → Control Flow Verification

Experimental Methodologies







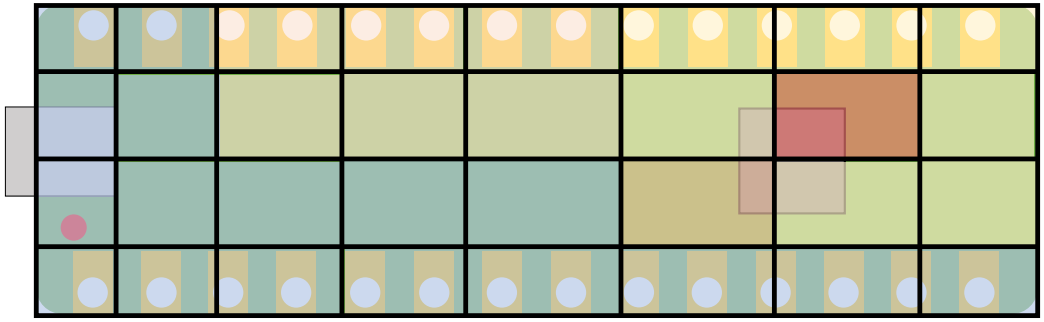


EM Leakage Detection → Basic Block Collection → Basic Block Classification → Control Flow Verification

1. Sampling Location Selection Process
2. Data Collection Harness
3. Basic Block Classification
4. Control Flow Decision

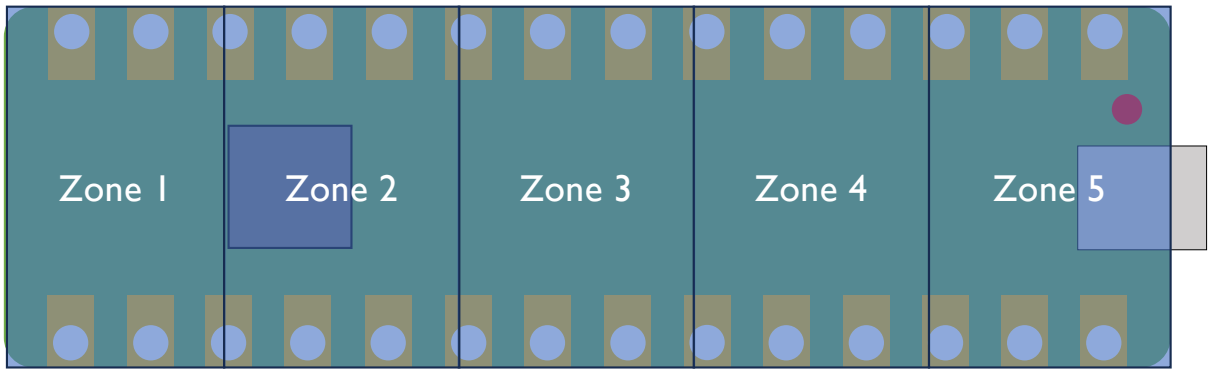
EM Leakage Detection

- Our initial experiments involved a CNC machine for automated sampling over an Arduino Uno [1]
- However, the same system did not exist for the Raspberry Pi Pico. We had to produce something new.

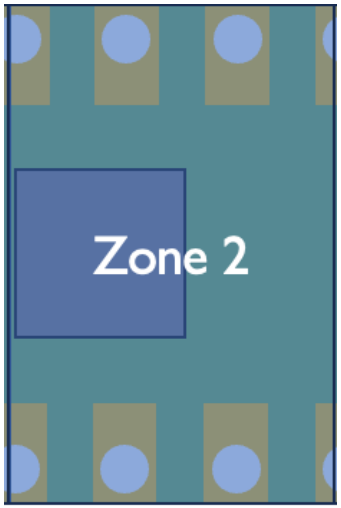


[1] Max Panoff, Ty Van Roy, and **Yier Jin**, "AutoEM: Automatic Trace Collection and Data Analysis for Electromagnetic Side Channel Security," *Government Microcircuit Applications and Critical Technology Conference (GOMACTech)*, 2021.

EM Leakage Detection







Automated vs Non-Automated Methodologies

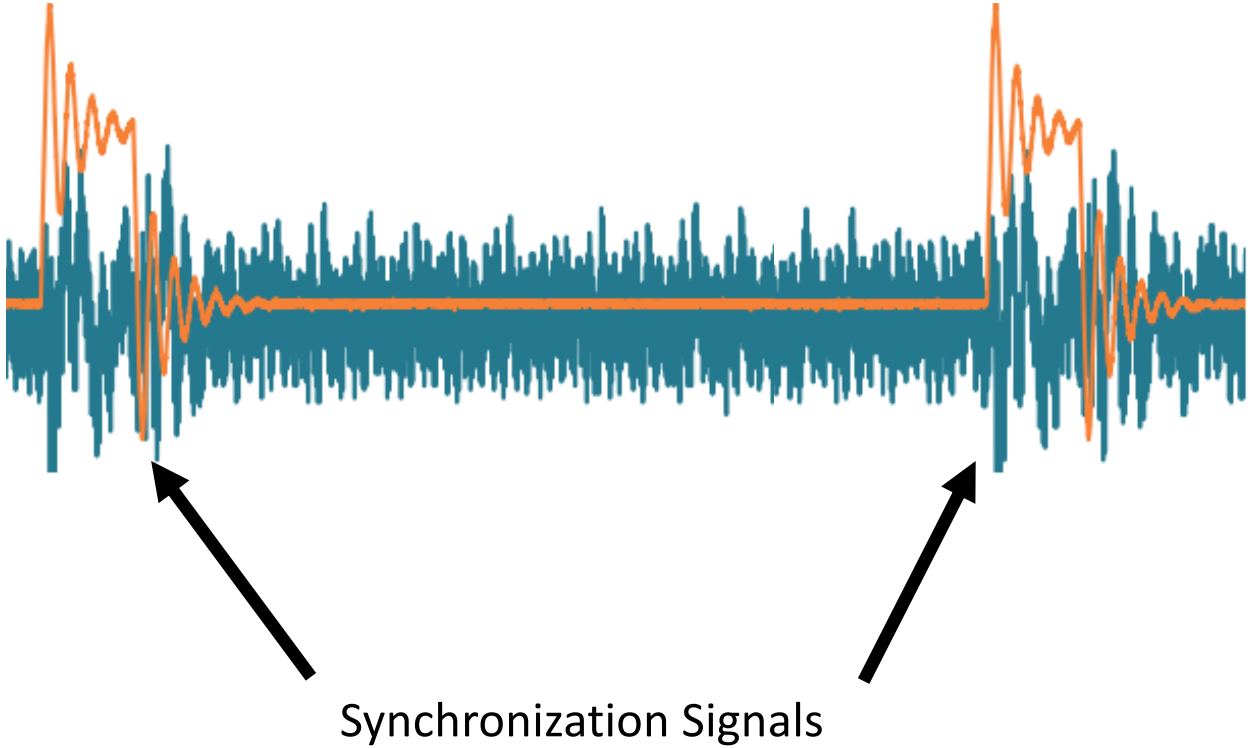
Automated Sampling

- Exhaustive; guarantees the best sampling position (on our test set)
- Expensive and time consuming to implement

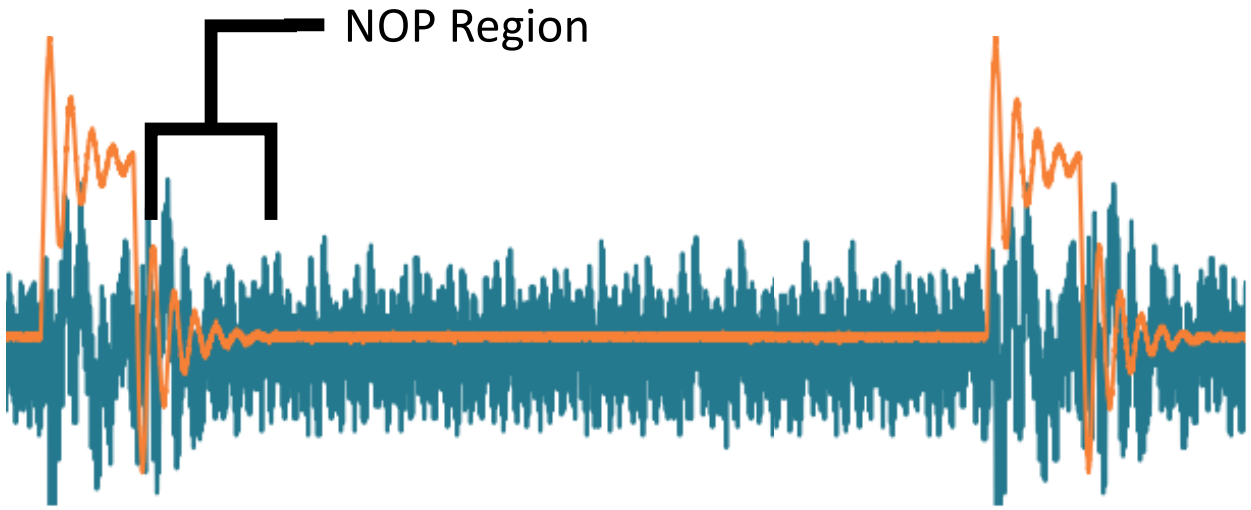
Non-Automated Sampling

- Non-exhaustive; may settle on a local minima
- Inexpensive and relatively few sequencing runs and data processing
- Works for any device or board

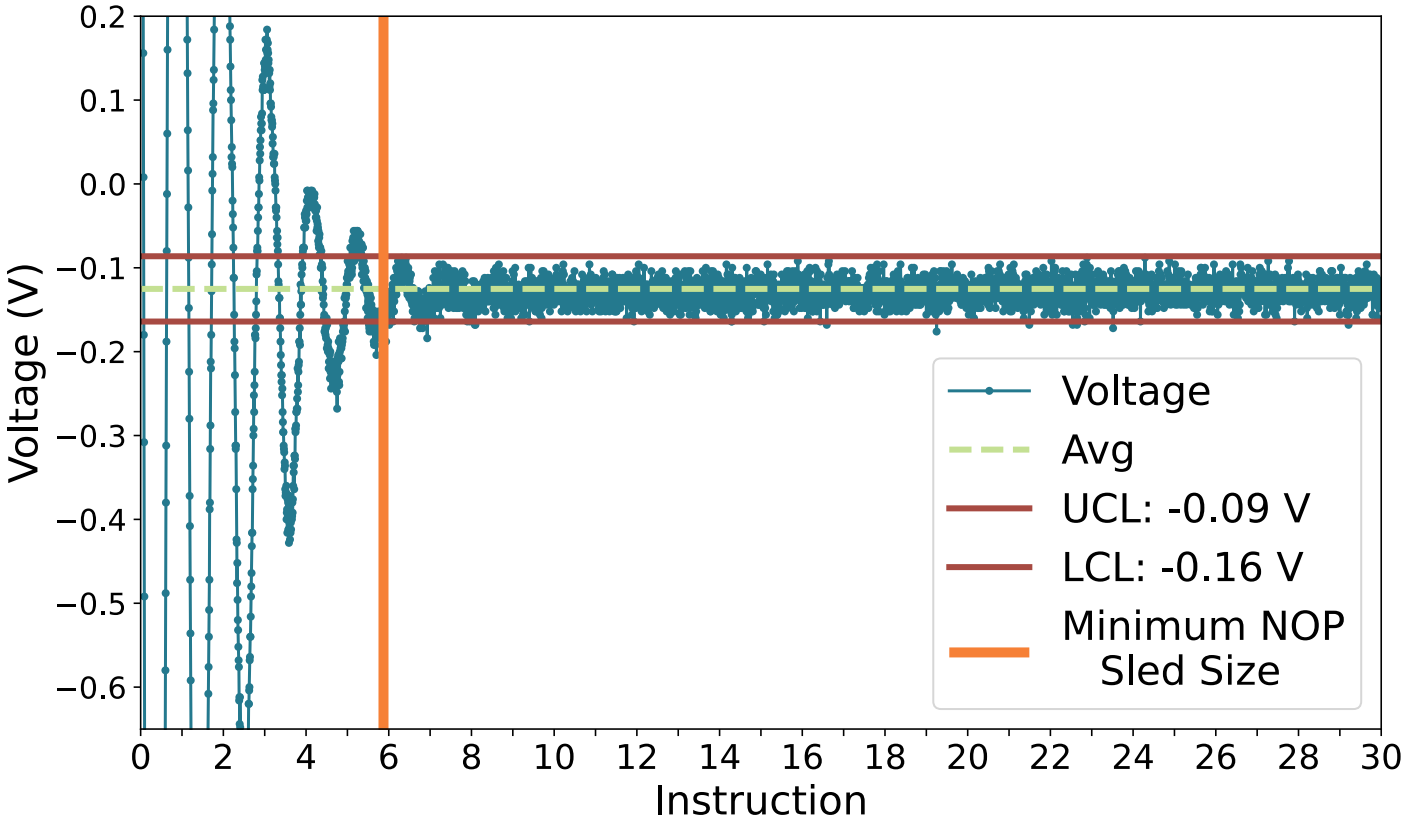
Synchronization Signals



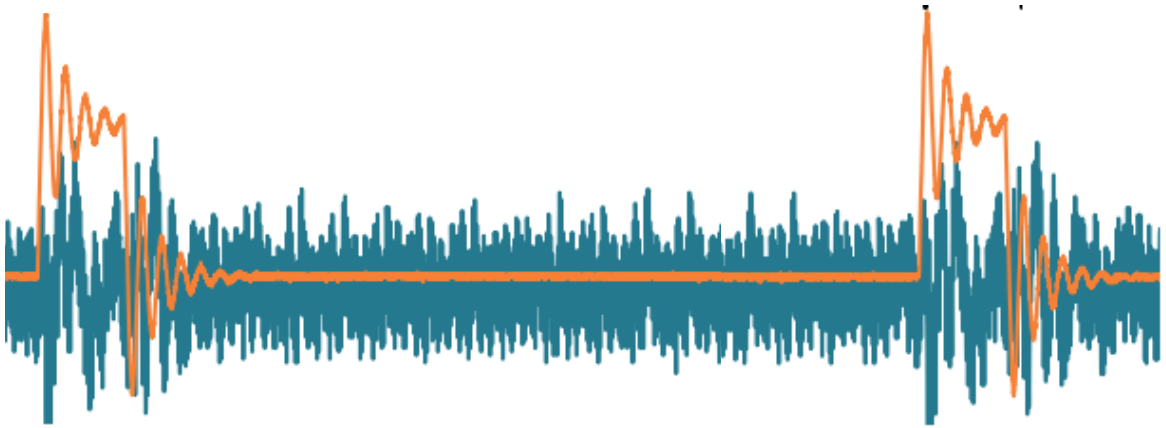
Synchronization Signals



Control Charts



Control Flow Graph



```
// Block A
ldi     r30, 0x0e
ldi     r31, 0x01
ldi     r26, 0x20
ldi     r27, 0x01

// Block B/C
ldi     r24, 0x01
cpi     r30, 0x1e
cpc     r31, r24
brne    .-6

// Block D
ldi     r24, 0x00
ret
```

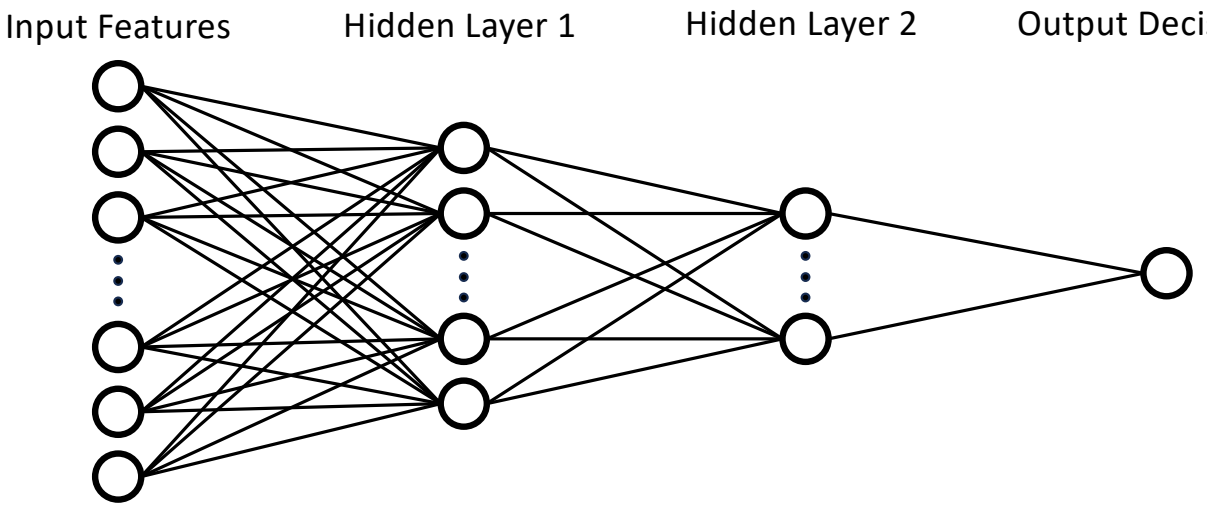
Selection of Relevant Programs

- We selected 6 programs for evaluation
  - Syringe Pump, Soldering Iron, Servo, Distance Sensor, Home IoT, Checksum
- There doesn't exist a robust set of programs to evaluate for embedded devices
  - Dalton list [1] is closest to standard, but many programs are outdated and inapplicable to the threat models we had

[1] Benchmark Applications for Synthesizeable VHDL Model. http://www.ann.ece.ufl.edu/i8051/i8051benchmarks/index.html Accessed: 2025-05-16.

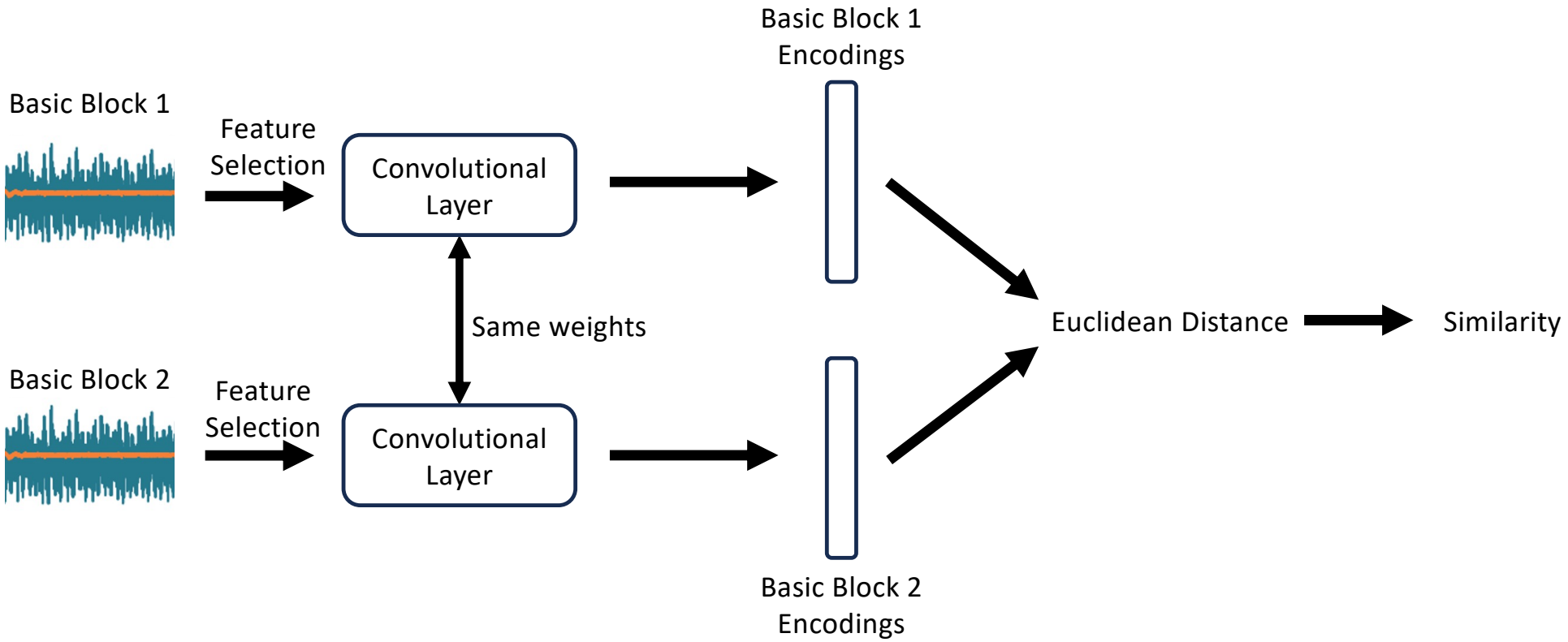
Basic Block Classification

- For block classification we selected between a number of different binary classification model architectures, selecting the model with the highest AUROC score

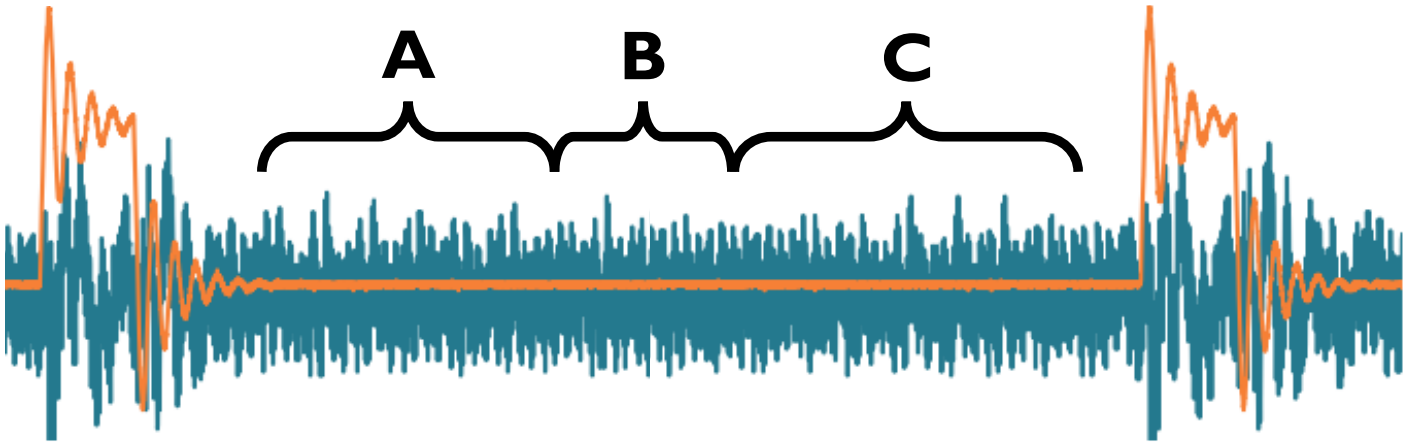


Basic Block Classification

- Since publication we have evaluated a Siamese Network Architecture
- This allows training a single model instead of individual classifiers



Control Flow Verification



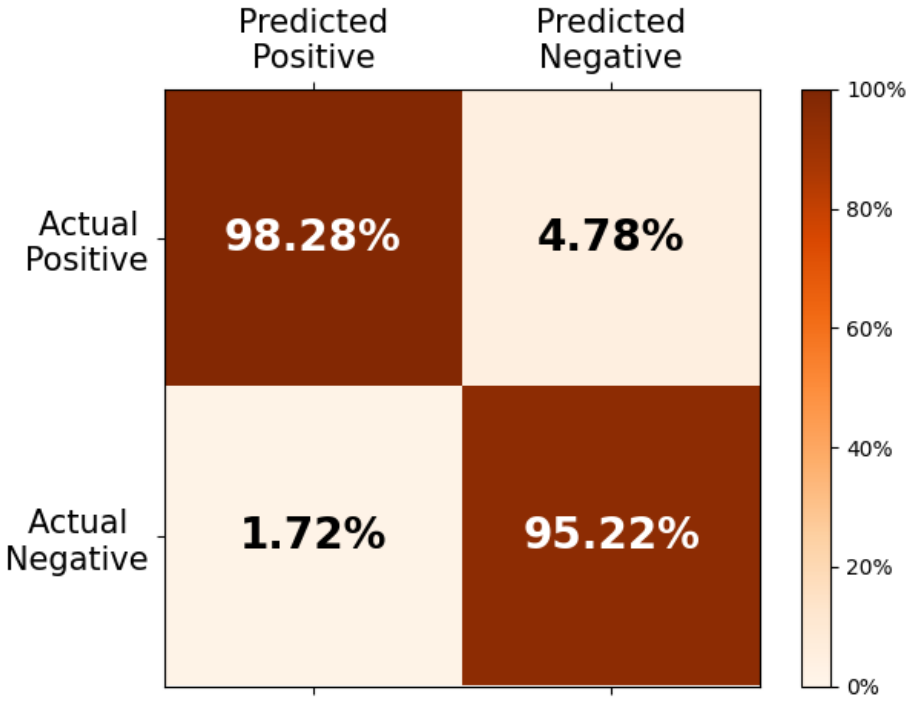
B-Score

- We combine these decisions together using a metric we define as the Benign-Score (b-score, β_μ)
- This is an average of all the classifier's accuracy across a single run
- We set the b-score threshold to yield a False Positive Rate of no more than 5%

$$\begin{cases} \boldsymbol{v_t} = (1 - |\boldsymbol{v_p} - \boldsymbol{v_c}|) \cdot \boldsymbol{a} \\ \beta_\mu = \dfrac{1}{n} \sum_{i=1}^{n} \boldsymbol{v_t} \end{cases}$$

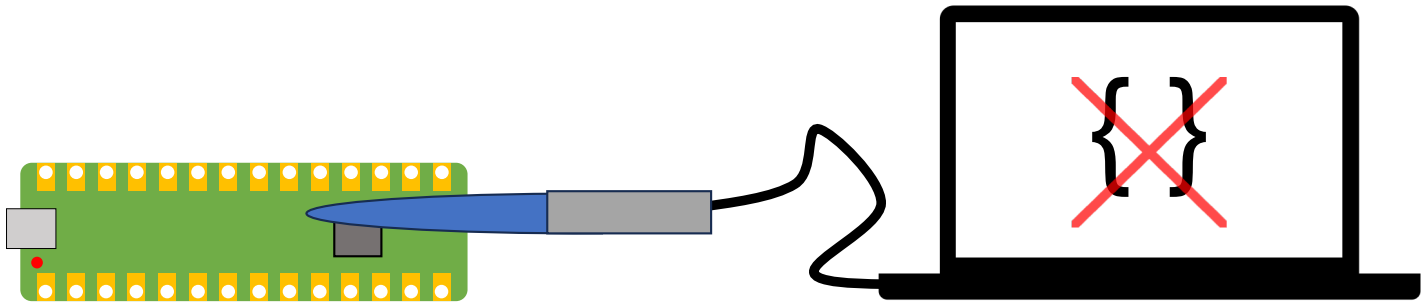
Performance Evaluation

- We focused on providing confusion matrix as opposed to overall accuracy in our evaluation
- Research in other areas has demonstrated how single metric focus can obscure performance



Reproduction of Prior Results

- Reproducibility in the EM space is still emerging
- None of the available papers open-sourced their code making direct reproduction difficult; we performed indirect comparison given their text

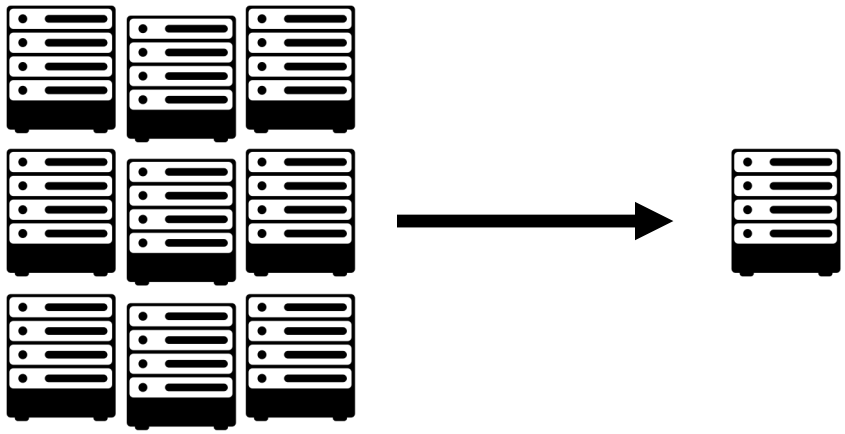


Artifact Evaluation Process

- At the end of our experiments, we had collected over 4 TB of data
- Training times for our models totaled near 24 hours per program
  - Translates to over a week of total training time on our hardware
- Evaluation using trained models takes considerably less time but depends on trace size
  - 15-20 mins per 300 executions

Artifact Evaluation Process

- We scoped our experiments
  - Provided trained models for evaluation
  - Included pre-processed data; segmented and feature selected (>200 GB)
  - Gave a small subset of complete traces to exercise the feature selection and training pipeline



Takeaways and Recommendations

- Docker-ized containers, run scripts, and discrete testable claims smoothed the review process
- Limitations exist in using common publicly accessible platforms especially for artifacts containing:
  - Large Datasets
  - Long execution times
- Transparent assignment processes with size and resources for AEC reviewers may ease evaluation

Thank you!

EM-Flow: Advanced Electromagnetic Control Flow Verification for Embedded Systems

