Recent Radiation Test Results on COTS AI Edge Processing ASICs

Megan Casey[1], Ed Wyrwas[2], and Rebekah Austin[1]

[1]NASA GSFC, Flight Data Systems and Radiation Effects Branch/Code 561

[2]SSAI, Inc, work performed for NASA GSFC

To be uploaded to nepp.nasa.gov as presented by Megan Casey at the NASA Electronic Parts and Packaging (NEPP) Program 2022 Electronics Technology Workshop, Greenbelt, MD, June 15 2022.

Acronyms

AI – Artificial Intelligence

ASIC – Application-Specific Integrated Circuit

COTS – Commercial Off the Shelf

CUVIS – Compact Ultraviolet to Visible Imaging Spectrometer

DAVINCI – Deep Atmosphere Venus Investigation of Noble gases, Chemistry, and Imaging

GPU – Graphics Processing Unit

LBNL – Lawrence Berkeley National Laboratory

MGH – Massachusetts General Hospital

NASA – National Aeronautics and Space Administration

NCS2 – Neural Compute Stick 2

NEPP – NASA Electronic Parts and Packaging Program

NSRL – NASA Space Radiation Laboratory

SEE – Single-Event Effects

SEFI – Single-Event Functional Interrupt

SEL – Single-Event Latchup

SEU – Single-Event Upset

TPU – Tensor Processing Unit

TSMC – Taiwan Semiconductor Manufacturing Company

VPU – Vision Processing Unit

Background

- COTS AI edge-processing ASICs have been on the NEPP radar for the past several years as part of the GPU subtask
  - In the fall of 2021, we were approached by members of the CUVIS instrument team (a technology demonstration on the DAVINCI mission) about conducting radiation testing of candidate AI edge-processing ASICs
- Parts tested were:
  - Google Coral Accelerator Module
    - Google Edge TPU
  - Intel Neural Compute Stick 2
    - Intel Movidius Myriad X VPU
- Both of these embedded solutions are specialized to perform AI inference in a small, low-power form factor and have recently been integrated into spaceflight platforms
  - They provide necessary AI compute capability for lower complexity algorithms while maintaining their size, weight, power, and cost requirements

Fabrication Process Nodes

- Intel Movidius Myriad X VPUs are manufactured in a 16 nm finFET process
- Google Edge TPU is thought to be manufactured in either 16 or 12 nm TSMC finFET process (unconfirmed)
  - Node sizes of Cloud TPUs are publicly available:
    - TPUv1 used 28-nm nodes
    - TPUv2 and TPUv3 used 16-nm nodes
    - TPUv4 used 7-nm nodes
  - TPUv2 and TPUv3 were released in the same timeframe as Edge TPU, it is likely the Edge TPU uses a similar 16-nm process
- Assuming Google TPUs are manufactured in either the 16 or 12 nm process, the parts would be expected to similar radiation response as the Intel VPUs

About the Models: Space-Based Applications

- **Space-based AI models focus on hyperspectral image classification**
  - **Inputs spectra to a multi-layer perceptron (MLP) classifier**
  - **Inputs both spectral and spatial information to a convolutional neural network (CNN)**
    - **Both models operated on the open-source Salinas hyperspectral dataset [14], estimating the land-usage class probabilities for each pixel in the image**
  - **Inputs planet's spectral information to a regression model consisting of a generative adversarial network (GAN) designed for exoplanet planetary atmospheric parameter retrieval (e.g., chemical species mixing ratio, temperature profile, or cloud properties)**
    - **This application was developed for the CUVIS instrument**
    - **It will enable researchers to analyze data on-board in near real time, to generate a reduced dataset to be returned in full, and to help flag and prioritize full resolution data to return**

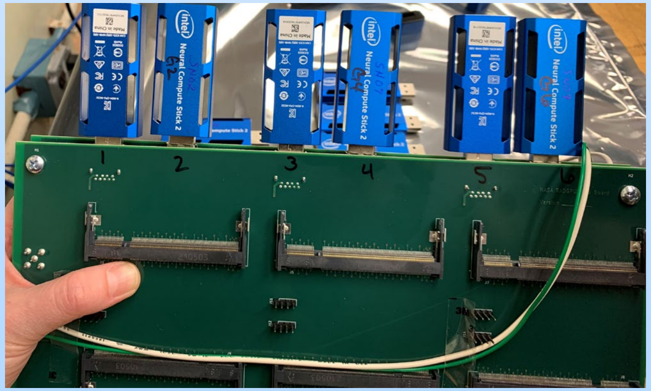
About the Models: Commercial Applications

- First of these commercial AI models is MobileNetV2, trained to perform image classification on the ImageNet dataset
  - Specifically designed for mobile applications, having significantly fewer parameters and operations compared to other state-of-the-art networks like ResNet50
- Second model is a Single-Shot Detector with a MobileNetV2 backbone (MobileNetV2-SSD) that is trained to perform object detection on the Common Objects in Context (COCO) dataset
  - Classifies multiple objects in an image and simultaneously estimates rectangular bounding boxes for where these objects occur in the image

Radiation Testing

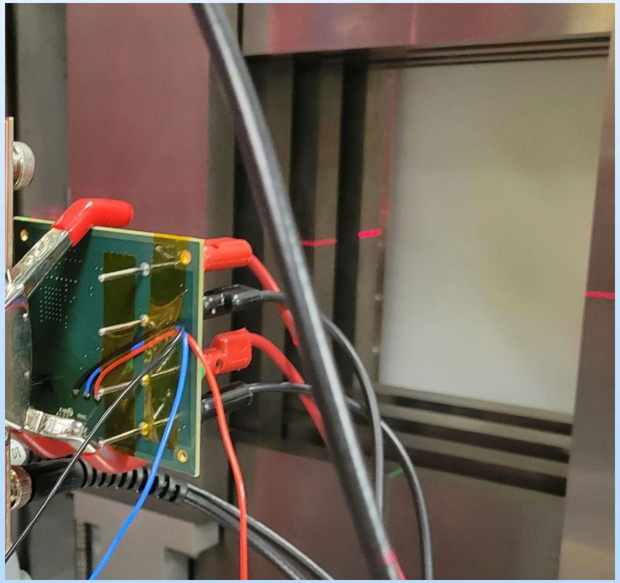
Total Ionizing Dose

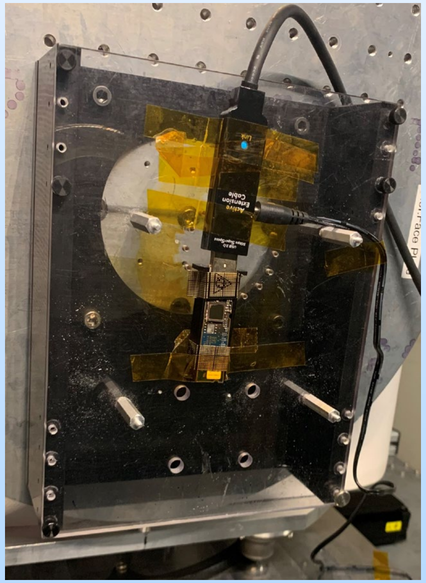
- Irradiated at GSFC's Radiation Effects Facility with 1.1-MeV gamma rays
  - Dose rates of 16.9 rad/s (Intel) and 15.8 rad/s (Google)
- 14 each of Intel and Google devices
  - 2 controls, 6 biased, and 6 unbiased



Single-Event Effects

- Heavy ion SEE irradiated at NSRL (Google) and LBNL (Intel)
- High-energy proton SEE irradiated at MGH

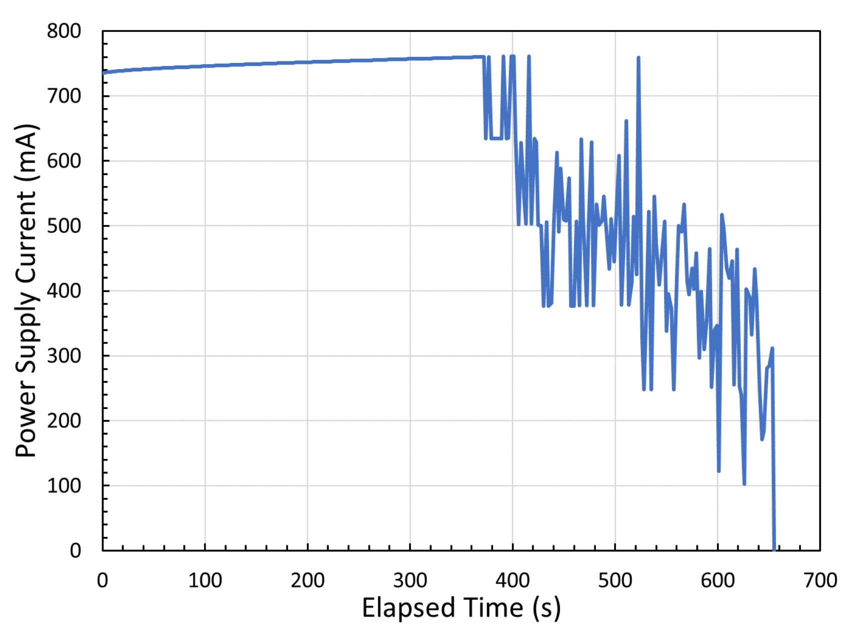




TID RESULTS

TID Results: Intel Neural Compute Stick 2

- No effects of TID were observed in any of the devices through 10 krad(Si)
  - Algorithm accuracies were unchanged indicating the logged output of the models were unchanged as a function of dose
- Between 10 and 20 krad(Si), different behaviors were observed in both biased and unbiased DUTs
  - In the biased devices, after approximately 6 krad(Si), the power supply current dropped 100s of mA
    - A single source was supplying voltage to all six DUTs, so the current is cumulative across all biased devices
    - The fluctuations varied by multiples of ~120-130 mA, which is the current draw of the individual devices
      - Indicates that some devices were failing intermittently *in situ*
  - However, the biased devices were able to connect to the host computer via USB and functioned normally
    - Further, all algorithm accuracies were still unchanged from the pre-irradiation values



TID Results: Intel Neural Compute Stick 2

- At the 20 krad(Si) dose point, the unbiased devices were put in the irradiation chamber in the Pb/Al boxes and biased for several minutes to determine if the current fluctuations observed in the biased devices may have been due to thermal changes
  - The supply current was steady however, so the parts were then grounded and the source was exposed to begin the irradiation.
- After the conclusion of that dose step, three of the unbiased DUTs failed functionally
  - They were unable to connect via USB and no current was drawing when the bias voltage was applied
- Surviving (both biased and unbiased) DUTs were all functioning normally (with the odd changes in current during irradiation in the biased DUTs) at smaller, incremental dose steps up to 25 krad(Si)

TID Results: Intel Neural Compute Stick 2

- Shortly after beginning the step to bring the biased devices to 25 krad(Si), an immediate decrease in the supply current was observed
  - Irradiation was paused and the devices were power cycled, but the current did not recover to the nominal level
  - Remained at the same lower current for remainder of step
  - After run was concluded, none of the biased devices could be connected to the host computer via USB – all six DUTs had failed functionally

TID Results: Intel NCS2 Conclusions

- Post-processing of raw data showed that no changes were observed in algorithm accuracies
- All device failures manifested as an inability to communicate over USB, indicating the USB controller portion of the circuit failed
- Post-irradiation failure analysis was conducted
  - Voltmeter indicated voltage regulator was functioning normally
  - Removing the voltage regulator and directly applying the supply voltage did not change the performance
- These are commercial devices with multiple commercial chips on them – Myriad X processor is not the limiting factor for TID performance of Intel Neural Compute Stick 2

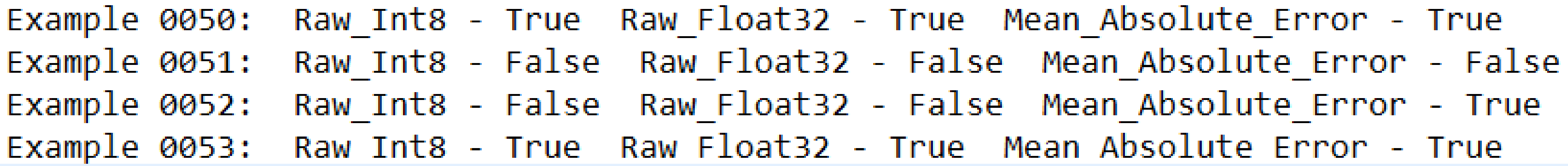
TID Results: Google Coral Edge TPU

- All DUTs (biased and unbiased) performed nominally up to 25 krad(Si)
- After 30 krad(Si), all 6 biased DUTs were unable to communicate over USB
  - PGOOD indicated on-chip voltage regulator was functioning as expected with the voltage measured at nominal 1.8 V
  - Algorithm accuracies were unchanged indicating logged output of the models were unchanged
- Unbiased devices were functioned normally with the same accuracies measured for all devices through 60 krad(Si)
- At 75 krad(Si), the unbiased DUTs failed similarly to the biased devices where they were unable to communicate via USB
  - PGOOD voltage was also nominal 1.8 V and algorithm accuracies were unchanged

SEE RESULTS

Observed SEE Signatures

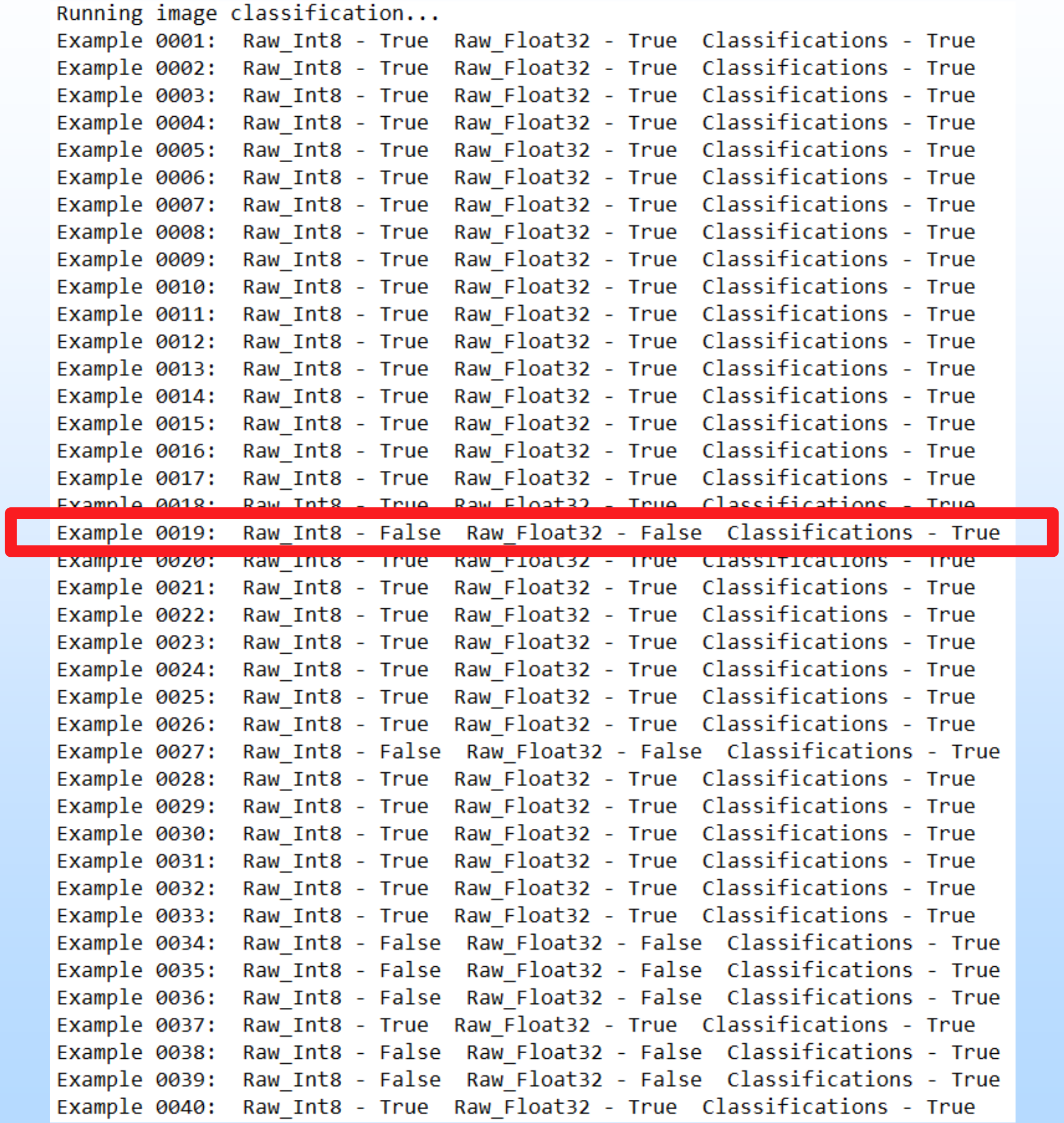
- **Types of SEEs observed:**
  - **SEUs manifested as changes in the algorithm accuracies that may or may not ultimately result in an error in the image classification**



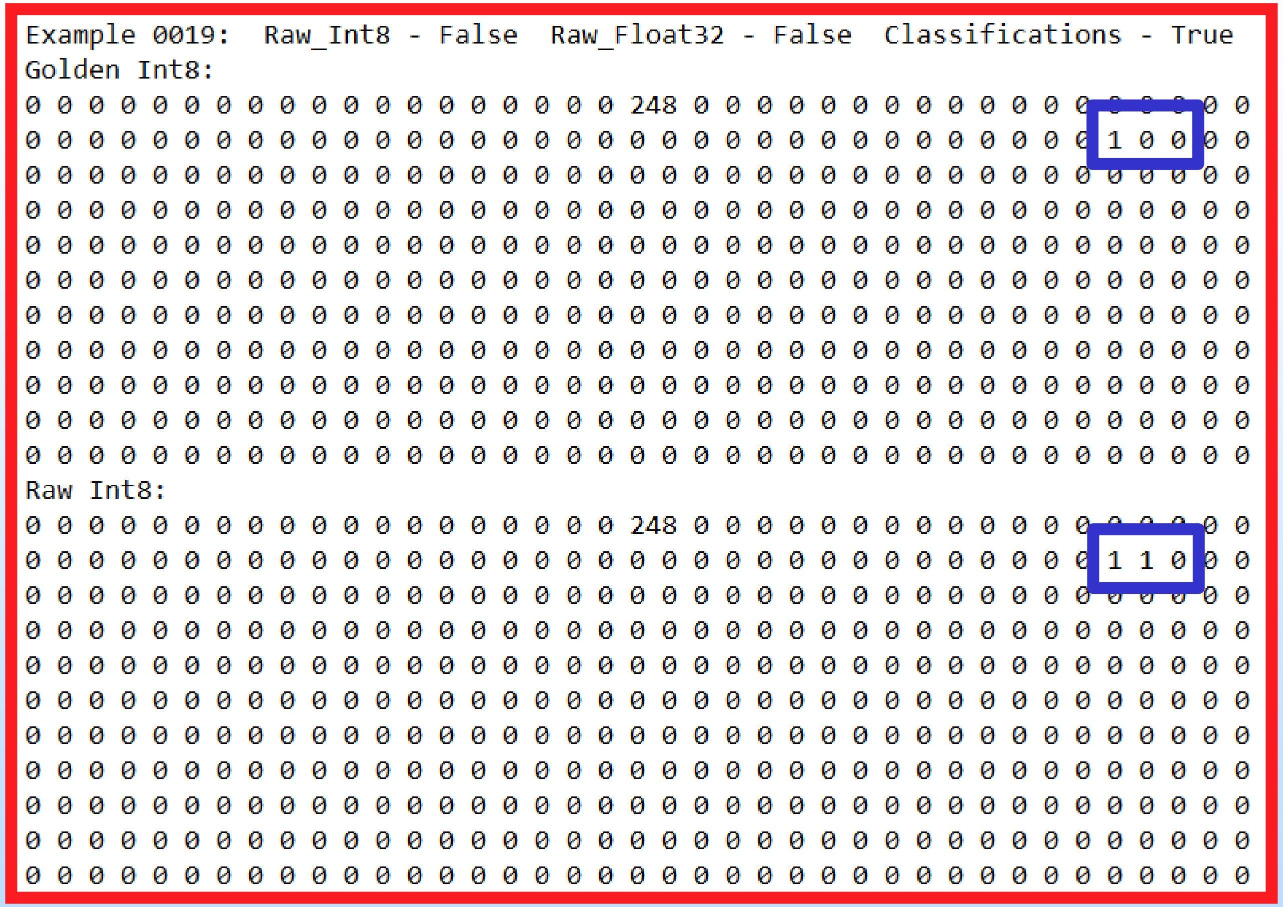
```
Example 0050:  Raw_Int8 - True  Raw_Float32 - True  Mean_Absolute_Error - True
Example 0051:  Raw_Int8 - False  Raw_Float32 - False  Mean_Absolute_Error - False
Example 0052:  Raw_Int8 - False  Raw_Float32 - False  Mean_Absolute_Error - True
Example 0053:  Raw Int8 - True  Raw Float32 - True  Mean Absolute Error - True
```

  - **SEFIs manifested as persistent errors in the algorithm accuracies that resulted in incorrect classifications, as an inability to communicate (device would “hang up”), or model would fail, but the program was able to move on to the next model**
  - **Stuck bits manifested as persistent Falses in the probability vectors, even after a power cycle**
    - **The stuck bits are most likely in the cache memory used for model weights or input data**

SEE Signatures: SEUs

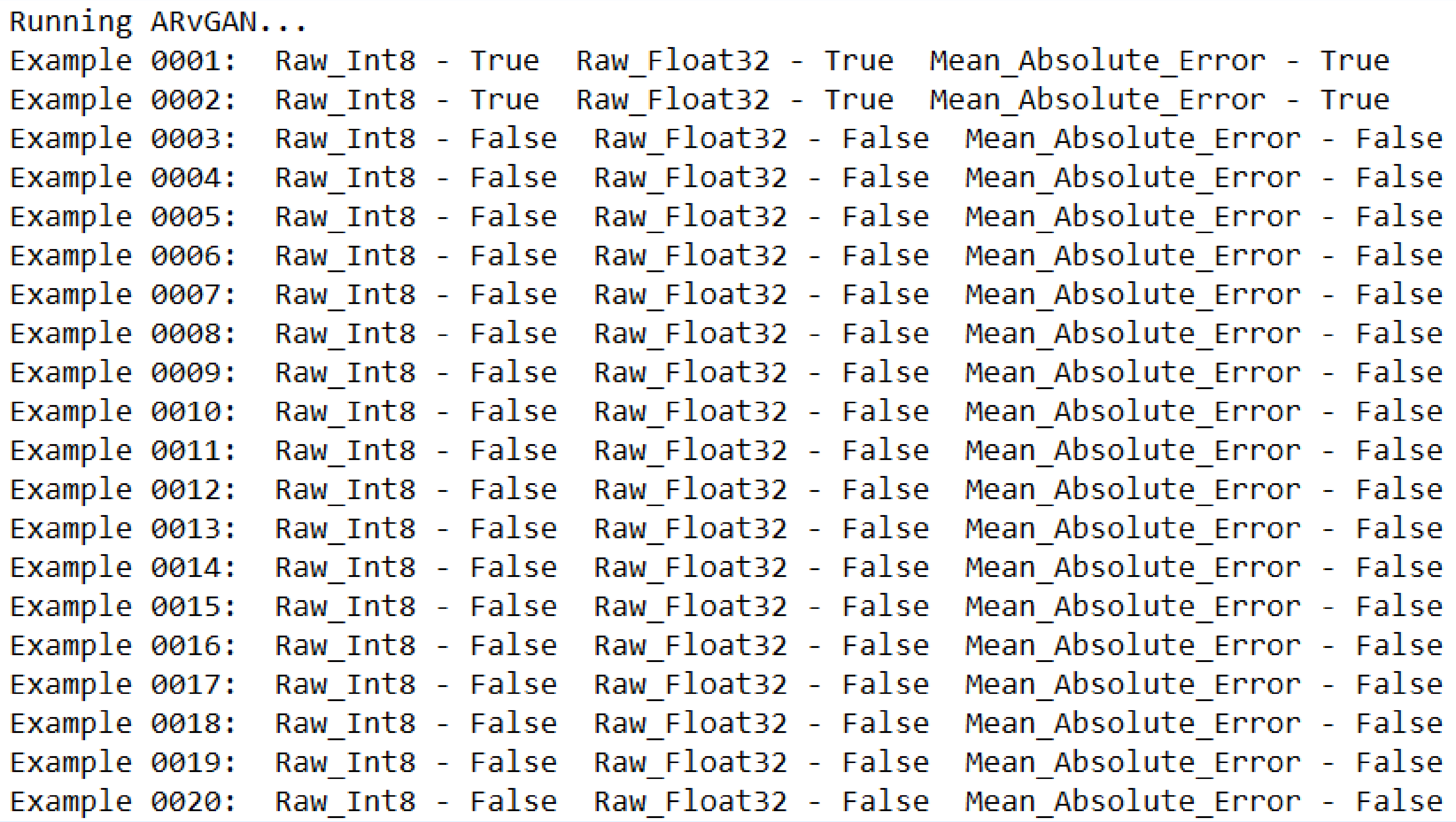


```
Running image classification...
Example 0001:  Raw_Int8 - True  Raw_Float32 - True  Classifications - True
Example 0002:  Raw_Int8 - True  Raw_Float32 - True  Classifications - True
Example 0003:  Raw_Int8 - True  Raw_Float32 - True  Classifications - True
Example 0004:  Raw_Int8 - True  Raw_Float32 - True  Classifications - True
Example 0005:  Raw_Int8 - True  Raw_Float32 - True  Classifications - True
Example 0006:  Raw_Int8 - True  Raw_Float32 - True  Classifications - True
Example 0007:  Raw_Int8 - True  Raw_Float32 - True  Classifications - True
Example 0008:  Raw_Int8 - True  Raw_Float32 - True  Classifications - True
Example 0009:  Raw_Int8 - True  Raw_Float32 - True  Classifications - True
Example 0010:  Raw_Int8 - True  Raw_Float32 - True  Classifications - True
Example 0011:  Raw_Int8 - True  Raw_Float32 - True  Classifications - True
Example 0012:  Raw_Int8 - True  Raw_Float32 - True  Classifications - True
Example 0013:  Raw_Int8 - True  Raw_Float32 - True  Classifications - True
Example 0014:  Raw_Int8 - True  Raw_Float32 - True  Classifications - True
Example 0015:  Raw_Int8 - True  Raw_Float32 - True  Classifications - True
Example 0016:  Raw_Int8 - True  Raw_Float32 - True  Classifications - True
Example 0017:  Raw_Int8 - True  Raw_Float32 - True  Classifications - True
Example 0018:  Raw_Int8 - True  Raw_Float32 - True  Classifications - True
Example 0019:  Raw_Int8 - False  Raw_Float32 - False  Classifications - True
Example 0020:  Raw_Int8 - True  Raw_Float32 - True  Classifications - True
Example 0021:  Raw_Int8 - True  Raw_Float32 - True  Classifications - True
Example 0022:  Raw_Int8 - True  Raw_Float32 - True  Classifications - True
Example 0023:  Raw_Int8 - True  Raw_Float32 - True  Classifications - True
Example 0024:  Raw_Int8 - True  Raw_Float32 - True  Classifications - True
Example 0025:  Raw_Int8 - True  Raw_Float32 - True  Classifications - True
Example 0026:  Raw_Int8 - True  Raw_Float32 - True  Classifications - True
Example 0027:  Raw_Int8 - False  Raw_Float32 - False  Classifications - True
Example 0028:  Raw_Int8 - True  Raw_Float32 - True  Classifications - True
Example 0029:  Raw_Int8 - True  Raw_Float32 - True  Classifications - True
Example 0030:  Raw_Int8 - True  Raw_Float32 - True  Classifications - True
Example 0031:  Raw_Int8 - True  Raw_Float32 - True  Classifications - True
Example 0032:  Raw_Int8 - True  Raw_Float32 - True  Classifications - True
Example 0033:  Raw_Int8 - True  Raw_Float32 - True  Classifications - True
Example 0034:  Raw_Int8 - False  Raw_Float32 - False  Classifications - True
Example 0035:  Raw_Int8 - False  Raw_Float32 - False  Classifications - True
Example 0036:  Raw_Int8 - False  Raw_Float32 - False  Classifications - True
Example 0037:  Raw_Int8 - True  Raw_Float32 - True  Classifications - True
Example 0038:  Raw_Int8 - False  Raw_Float32 - False  Classifications - True
Example 0039:  Raw_Int8 - False  Raw_Float32 - False  Classifications - True
Example 0040:  Raw_Int8 - True  Raw_Float32 - True  Classifications - True
```



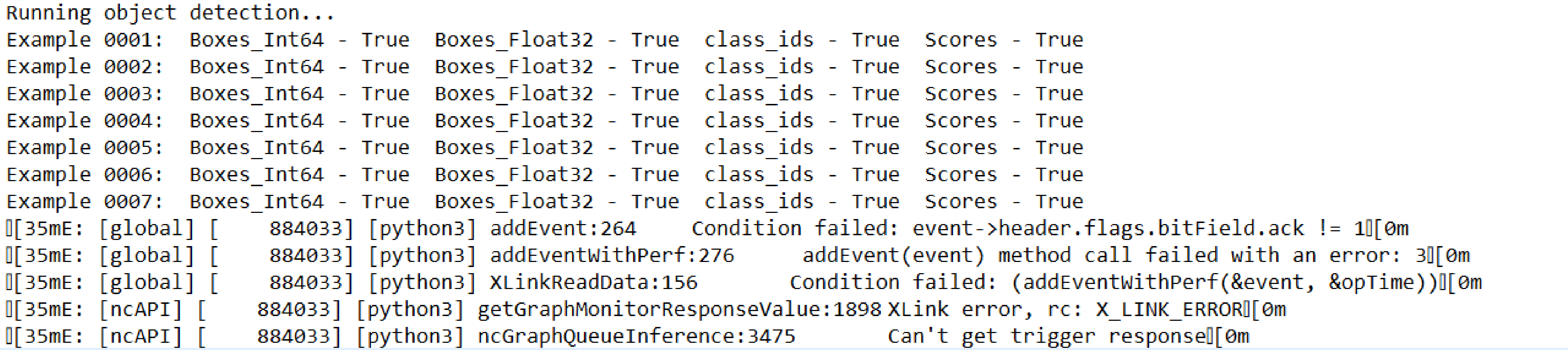
```
Example 0019:  Raw_Int8 - False  Raw_Float32 - False  Classifications - True
Golden Int8:
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 248 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
Raw Int8:
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 248 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

SEE Signatures: Recoverable SEFIs



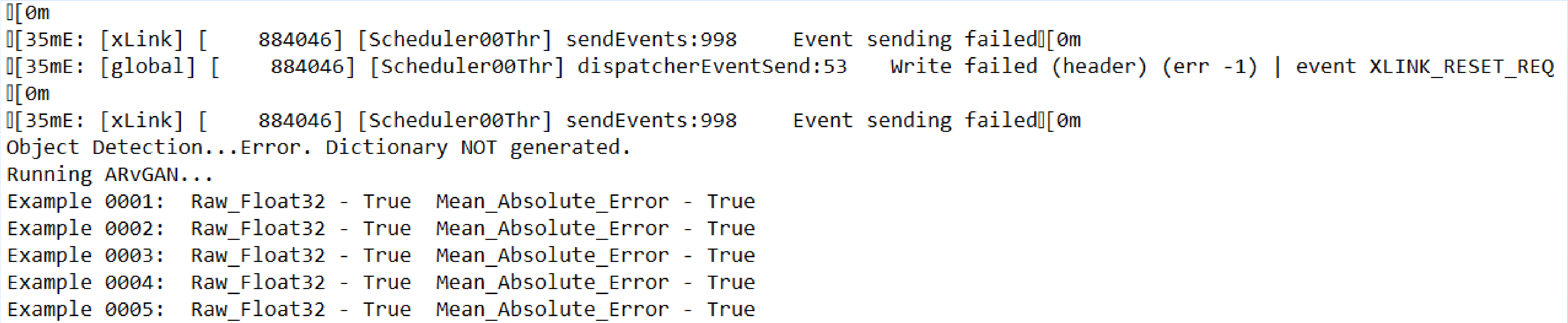
```
Running ARvGAN...
Example 0001:  Raw_Int8 - True  Raw_Float32 - True  Mean_Absolute_Error - True
Example 0002:  Raw_Int8 - True  Raw_Float32 - True  Mean_Absolute_Error - True
Example 0003:  Raw_Int8 - False  Raw_Float32 - False  Mean_Absolute_Error - False
Example 0004:  Raw_Int8 - False  Raw_Float32 - False  Mean_Absolute_Error - False
Example 0005:  Raw_Int8 - False  Raw_Float32 - False  Mean_Absolute_Error - False
Example 0006:  Raw_Int8 - False  Raw_Float32 - False  Mean_Absolute_Error - False
Example 0007:  Raw_Int8 - False  Raw_Float32 - False  Mean_Absolute_Error - False
Example 0008:  Raw_Int8 - False  Raw_Float32 - False  Mean_Absolute_Error - False
Example 0009:  Raw_Int8 - False  Raw_Float32 - False  Mean_Absolute_Error - False
Example 0010:  Raw_Int8 - False  Raw_Float32 - False  Mean_Absolute_Error - False
Example 0011:  Raw_Int8 - False  Raw_Float32 - False  Mean_Absolute_Error - False
Example 0012:  Raw_Int8 - False  Raw_Float32 - False  Mean_Absolute_Error - False
Example 0013:  Raw_Int8 - False  Raw_Float32 - False  Mean_Absolute_Error - False
Example 0014:  Raw_Int8 - False  Raw_Float32 - False  Mean_Absolute_Error - False
Example 0015:  Raw_Int8 - False  Raw_Float32 - False  Mean_Absolute_Error - False
Example 0016:  Raw_Int8 - False  Raw_Float32 - False  Mean_Absolute_Error - False
Example 0017:  Raw_Int8 - False  Raw_Float32 - False  Mean_Absolute_Error - False
Example 0018:  Raw_Int8 - False  Raw_Float32 - False  Mean_Absolute_Error - False
Example 0019:  Raw_Int8 - False  Raw_Float32 - False  Mean_Absolute_Error - False
Example 0020:  Raw_Int8 - False  Raw_Float32 - False  Mean_Absolute_Error - False
```

SEE Signatures: Recoverable SEFIs



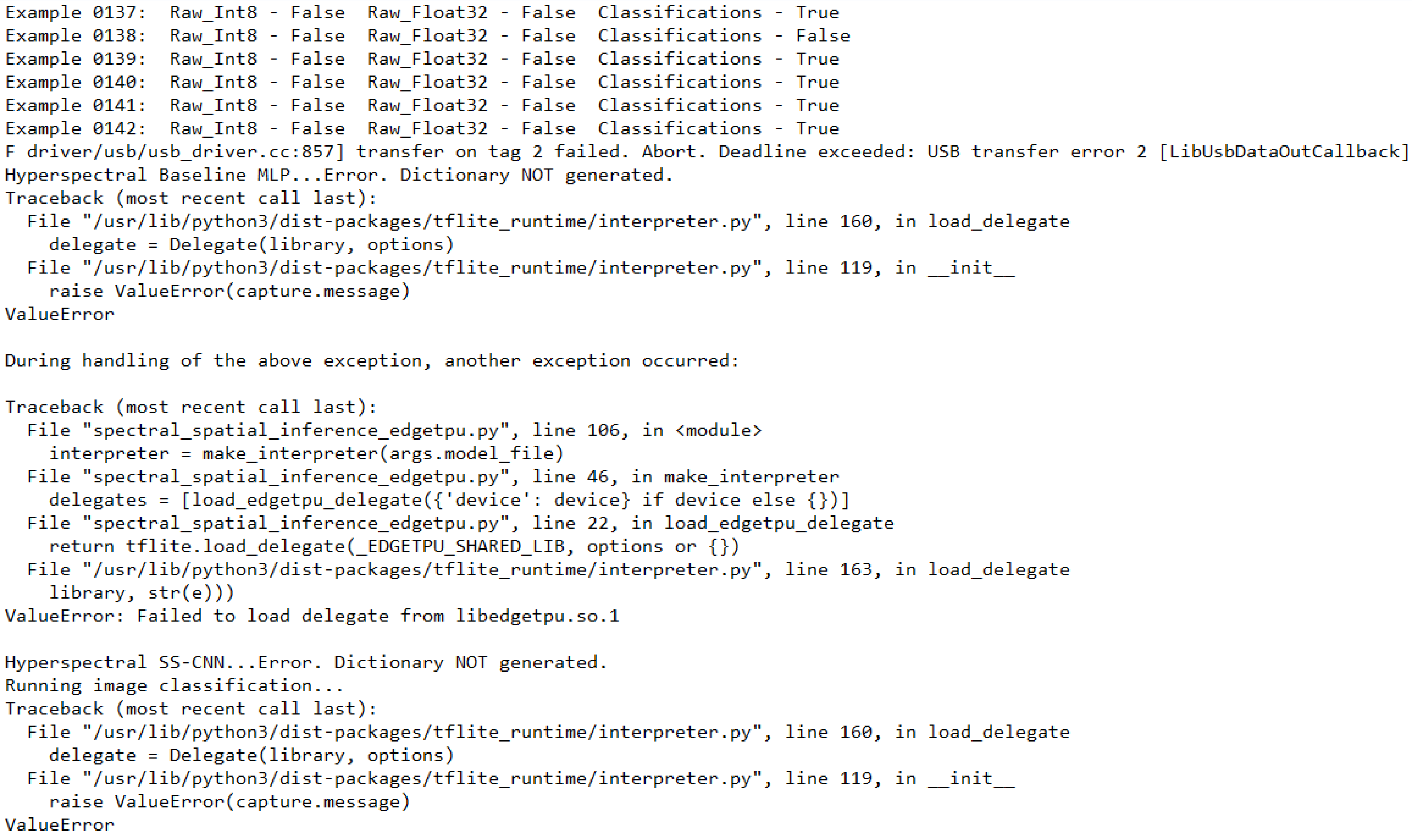
```
Running object detection...
Example 0001:  Boxes_Int64 - True  Boxes_Float32 - True  class_ids - True  Scores - True
Example 0002:  Boxes_Int64 - True  Boxes_Float32 - True  class_ids - True  Scores - True
Example 0003:  Boxes_Int64 - True  Boxes_Float32 - True  class_ids - True  Scores - True
Example 0004:  Boxes_Int64 - True  Boxes_Float32 - True  class_ids - True  Scores - True
Example 0005:  Boxes_Int64 - True  Boxes_Float32 - True  class_ids - True  Scores - True
Example 0006:  Boxes_Int64 - True  Boxes_Float32 - True  class_ids - True  Scores - True
Example 0007:  Boxes_Int64 - True  Boxes_Float32 - True  class_ids - True  Scores - True
▯[35mE: [global] [    884033] [python3] addEvent:264    Condition failed: event->header.flags.bitField.ack != 1▯[0m
▯[35mE: [global] [    884033] [python3] addEventWithPerf:276     addEvent(event) method call failed with an error: 3▯[0m
▯[35mE: [global] [    884033] [python3] XLinkReadData:156       Condition failed: (addEventWithPerf(&event, &opTime))▯[0m
▯[35mE: [ncAPI] [    884033] [python3] getGraphMonitorResponseValue:1898 XLink error, rc: X_LINK_ERROR▯[0m
▯[35mE: [ncAPI] [    884033] [python3] ncGraphQueueInference:3475       Can't get trigger response▯[0m
```

●   ●   ●
●   ●   ●
●   ●   ●



```
▯[0m
▯[35mE: [xLink] [    884046] [Scheduler00Thr] sendEvents:998    Event sending failed▯[0m
▯[35mE: [global] [    884046] [Scheduler00Thr] dispatcherEventSend:53    Write failed (header) (err -1) | event XLINK_RESET_REQ
▯[0m
▯[35mE: [xLink] [    884046] [Scheduler00Thr] sendEvents:998    Event sending failed▯[0m
Object Detection...Error. Dictionary NOT generated.
Running ARvGAN...
Example 0001:  Raw_Float32 - True  Mean_Absolute_Error - True
Example 0002:  Raw_Float32 - True  Mean_Absolute_Error - True
Example 0003:  Raw_Float32 - True  Mean_Absolute_Error - True
Example 0004:  Raw_Float32 - True  Mean_Absolute_Error - True
Example 0005:  Raw_Float32 - True  Mean_Absolute_Error - True
```

SEE Signatures: Non-recoverable SEFIs



```
Example 0137:  Raw_Int8 - False  Raw_Float32 - False  Classifications - True
Example 0138:  Raw_Int8 - False  Raw_Float32 - False  Classifications - False
Example 0139:  Raw_Int8 - False  Raw_Float32 - False  Classifications - True
Example 0140:  Raw_Int8 - False  Raw_Float32 - False  Classifications - True
Example 0141:  Raw_Int8 - False  Raw_Float32 - False  Classifications - True
Example 0142:  Raw_Int8 - False  Raw_Float32 - False  Classifications - True
F driver/usb/usb_driver.cc:857] transfer on tag 2 failed. Abort. Deadline exceeded: USB transfer error 2 [LibUsbDataOutCallback]
Hyperspectral Baseline MLP...Error. Dictionary NOT generated.
Traceback (most recent call last):
  File "/usr/lib/python3/dist-packages/tflite_runtime/interpreter.py", line 160, in load_delegate
    delegate = Delegate(library, options)
  File "/usr/lib/python3/dist-packages/tflite_runtime/interpreter.py", line 119, in __init__
    raise ValueError(capture.message)
ValueError

During handling of the above exception, another exception occurred:

Traceback (most recent call last):
  File "spectral_spatial_inference_edgetpu.py", line 106, in <module>
    interpreter = make_interpreter(args.model_file)
  File "spectral_spatial_inference_edgetpu.py", line 46, in make_interpreter
    delegates = [load_edgetpu_delegate({'device': device} if device else {})]
  File "spectral_spatial_inference_edgetpu.py", line 22, in load_edgetpu_delegate
    return tflite.load_delegate(_EDGETPU_SHARED_LIB, options or {})
  File "/usr/lib/python3/dist-packages/tflite_runtime/interpreter.py", line 163, in load_delegate
    library, str(e)))
ValueError: Failed to load delegate from libedgetpu.so.1

Hyperspectral SS-CNN...Error. Dictionary NOT generated.
Running image classification...
Traceback (most recent call last):
  File "/usr/lib/python3/dist-packages/tflite_runtime/interpreter.py", line 160, in load_delegate
    delegate = Delegate(library, options)
  File "/usr/lib/python3/dist-packages/tflite_runtime/interpreter.py", line 119, in __init__
    raise ValueError(capture.message)
ValueError
```

SEE Results: Heavy Ions

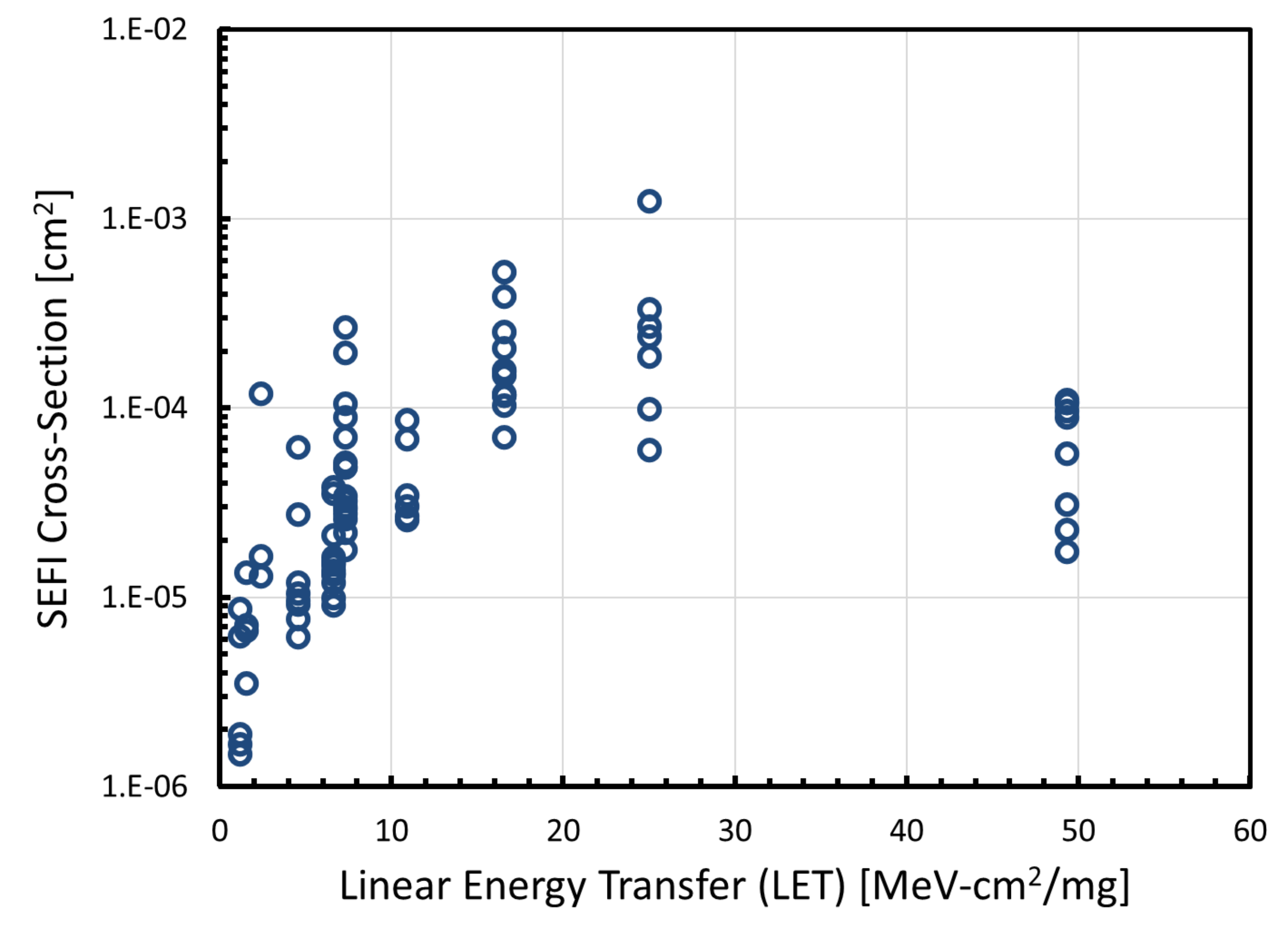
Intel Neural Compute Stick 2

- Myriad X was decapsulated and NCS2 was tested at LBNL
- No SEL observed at max tested LET of 49.3 MeV-cm^2/mg
- SEU and SEFI LET$_{th}$ ~ 1.16 MeV-cm^2/mg
  - No lighter ions available in 16 MeV/n beam tune
- No stuck bits observed

Google Coral Edge TPU

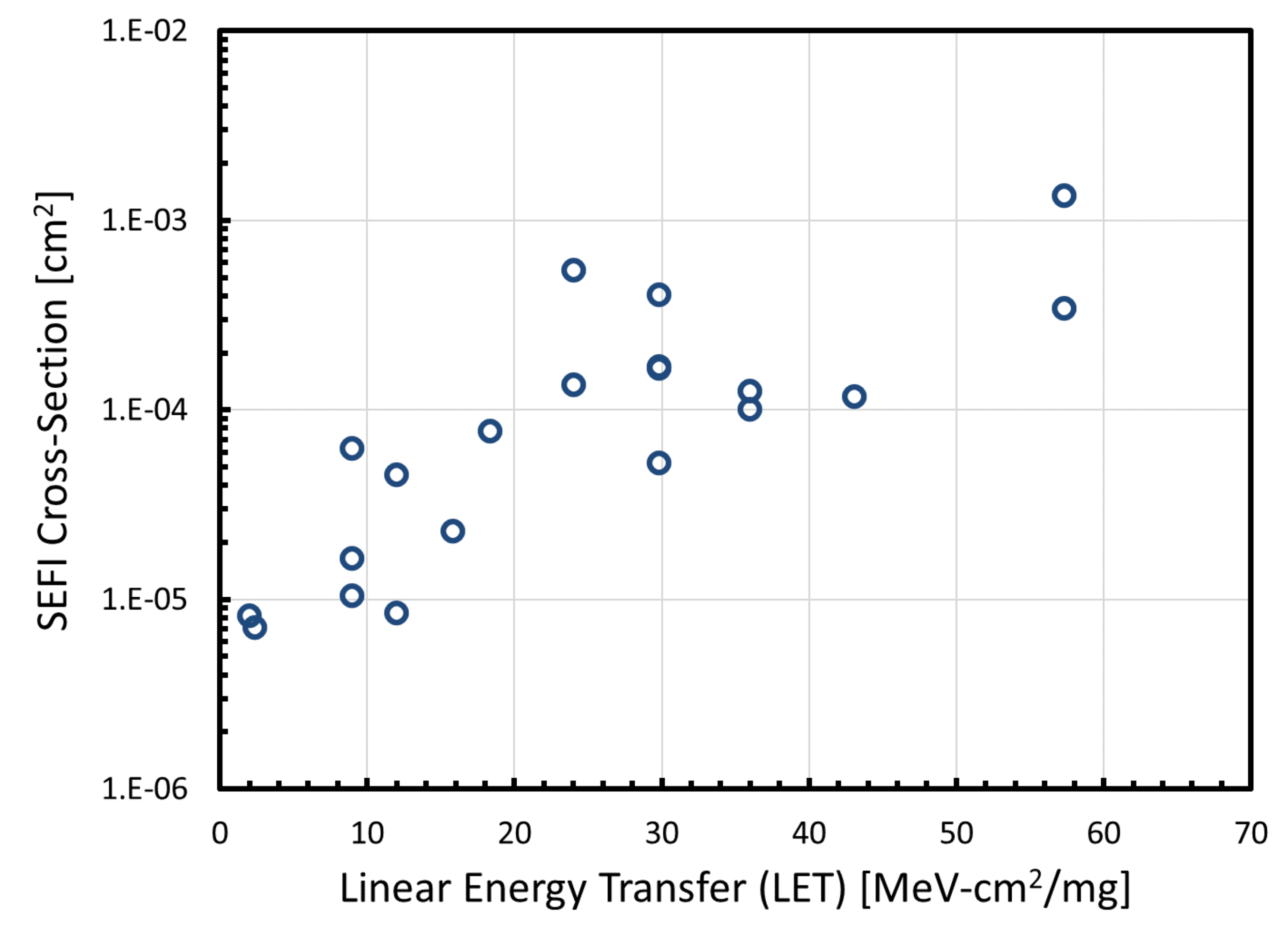
- Google Coral TPU could not be decapsulated so devices were tested at NSRL
- No SEL observed at max tested LET of 57.3 MeV-cm^2/mg
- SEU and SEFI LET$_{th}$ ~ 1.96 MeV-cm^2/mg
  - No SEEs were observed at 0.5 MeV-cm^2/mg
- Stuck bits were observed at LET of 57.3 MeV-cm^2/mg

Heavy Ion SEFI Cross-section: Intel NCS2



For geosynchronous/interplanetary mission during solar minimum, expected rate is 0.083 SEFIs/day

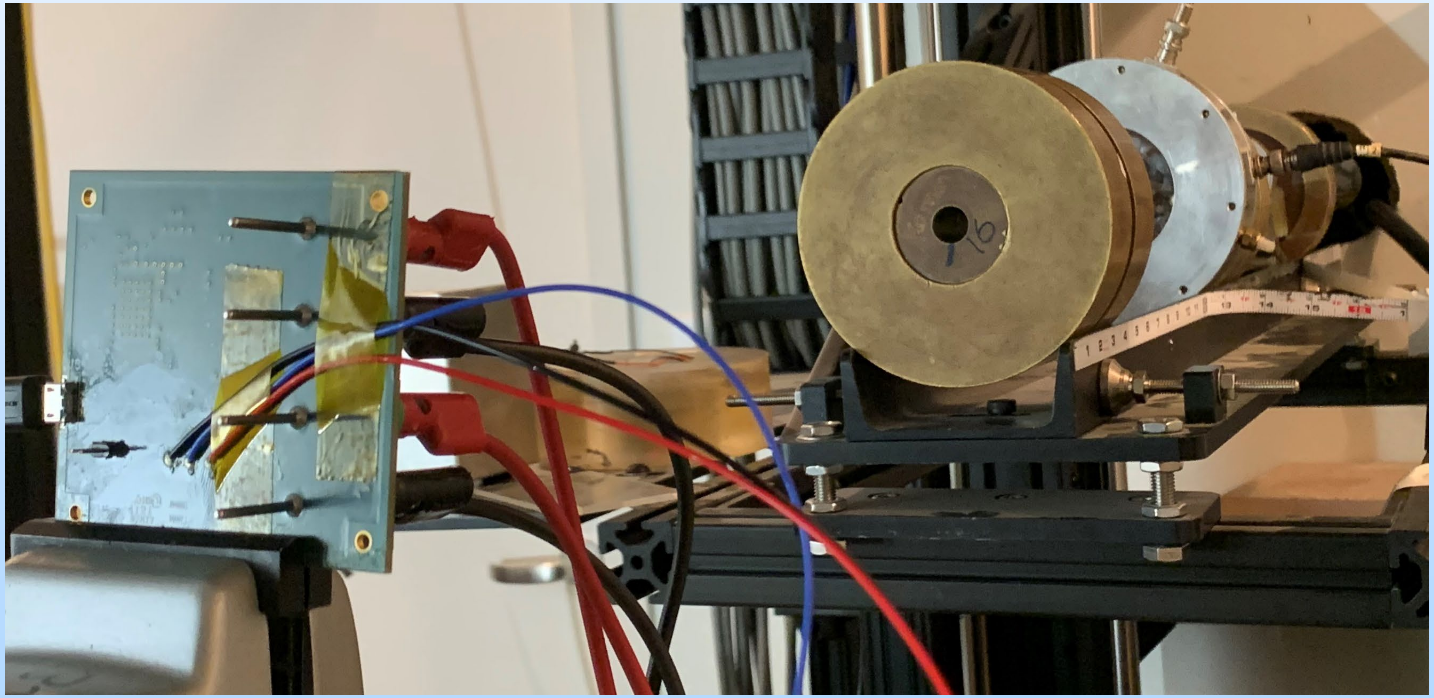
Heavy Ion SEFI Cross-section: Google Edge TPU



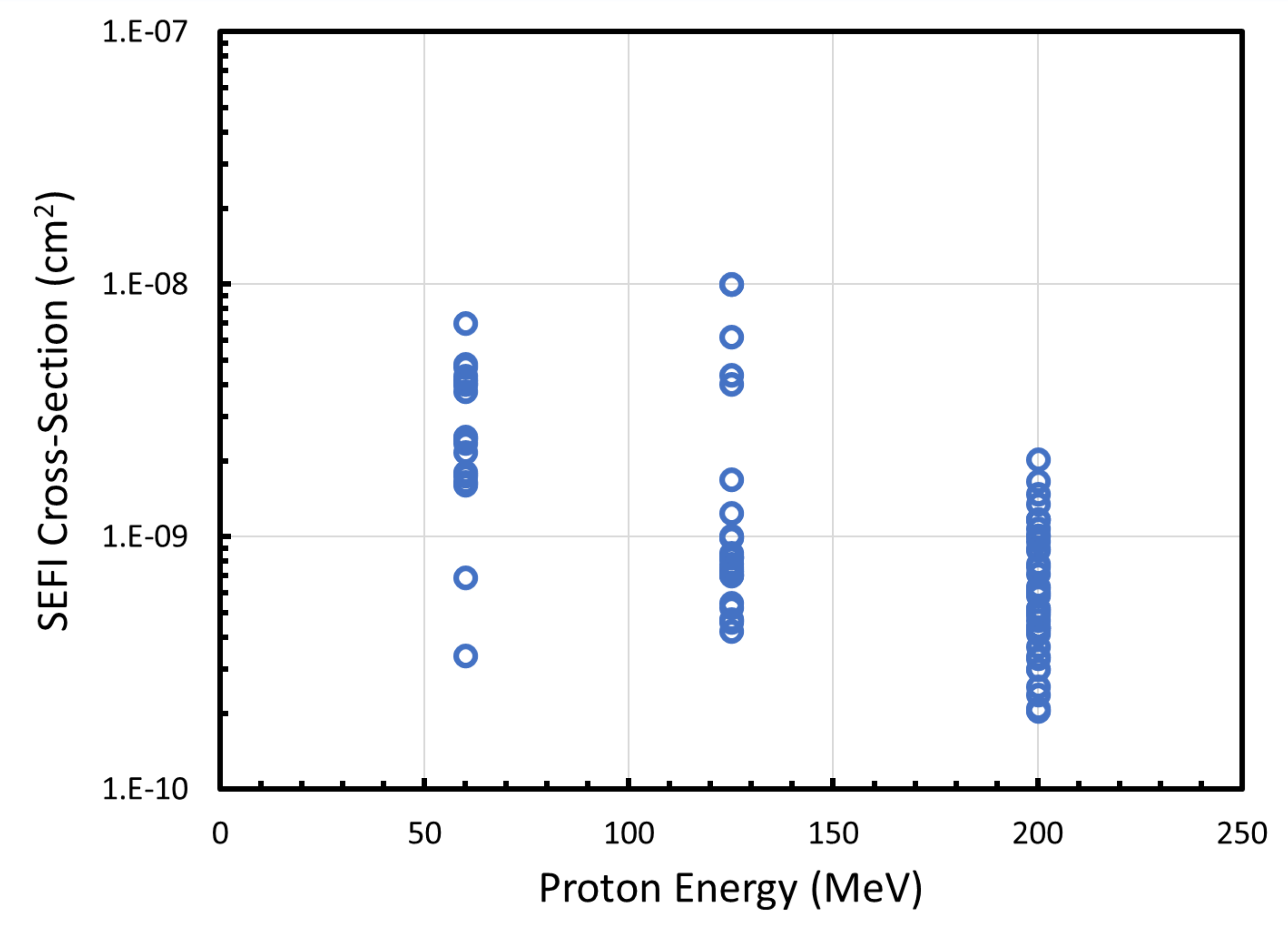
For geosynchronous/interplanetary mission during solar minimum, expected rate is 0.017 SEFIs/day

SEE Results: High-Energy Protons

- Both devices were tested at MGH with 60-, 125-, and 200-MeV protons
- No SEL or stuck bits observed
- Same SEU and SEFI signatures were observed as when irradiated with heavy ions

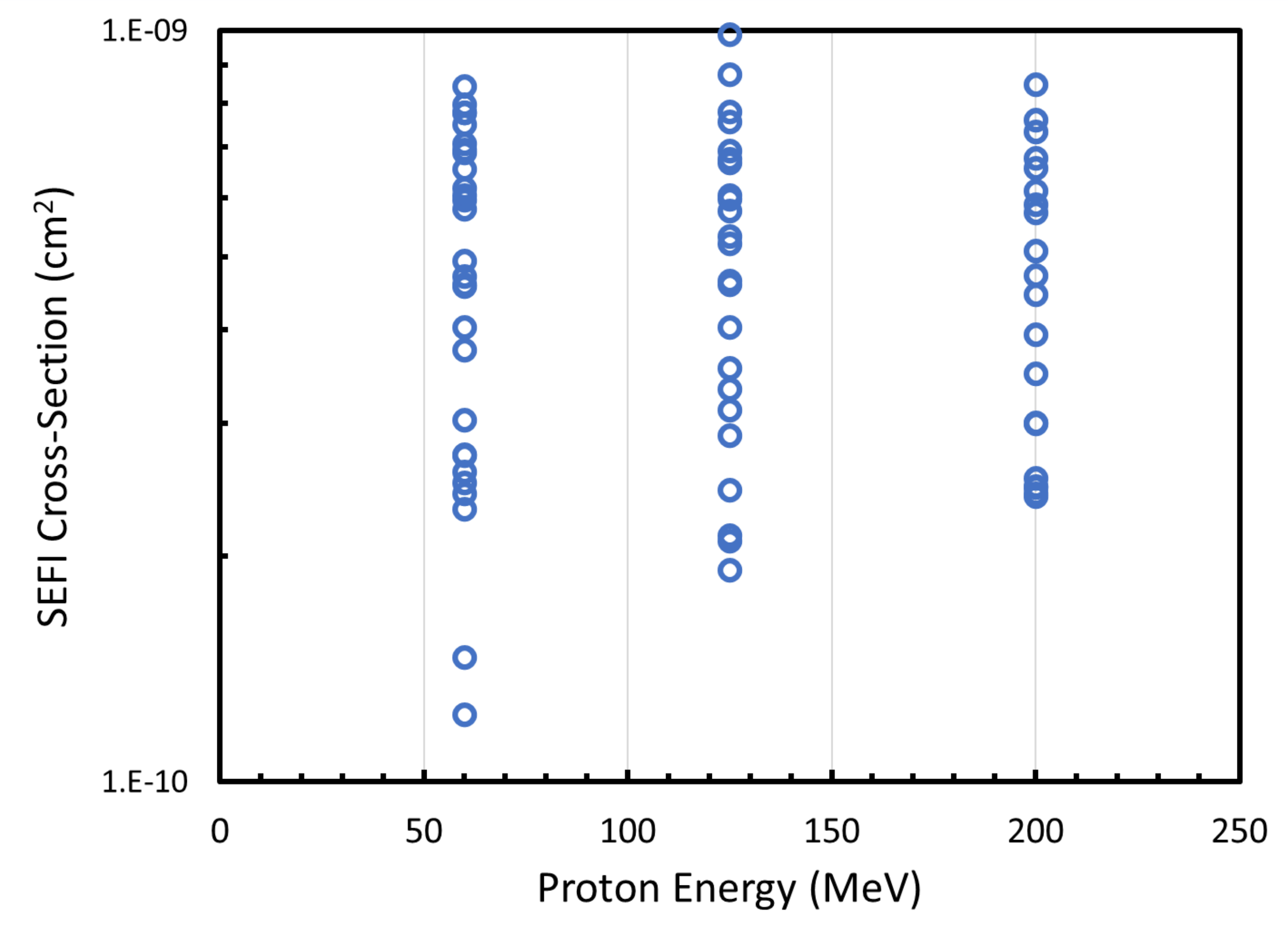


Proton SEFI Cross-section: Intel NCS2



For geosynchronous/interplanetary mission during solar minimum, the proton contribution to the expected rate is 0.0035 SEFIs/day

Proton SEFI Cross-section: Google Edge TPU



For geosynchronous/interplanetary mission during solar minimum, the proton contribution to the expected rate is 0.00035 SEFIs/day

SEE Conclusions

- Implementation of AI models resulted in very similar SEE signatures and behavior in Intel NCS2 and Google Edge TPU
    - Block diagrams of the devices are limited, so exact cause/sensitive location of each type is unknown
- Heavy-ion-induced SEFI cross-section of each device type is similar
    - However, there appears to have been a range issue at the highest tested LET in the Intel NCS2s, so other types of SEEs may be possible in these devices
- Same SEE signatures were observed with high-energy protons as with heavy ions
- Approximately an order of magnitude higher proton cross-sections with Intel NCS2s than Google Edge TPU
    - Results in proportional increase in expected number of SEFIs due to protons