

HLS for Handwritten Digits Recognition

Group Number: **18**

Group members with Roll Number:

234101001	Abhishek Pandey
234101002	Abhishek Verma
234101009	Aman Agrawal
234101011	Brajesh Kar
234101052	Susmit Das

The final report that needs to be submitted should contain the following:

1. Description of the model. Please write briefly about the ML model given to you along with the following data.

Model Description

The given Convolutional Neural Network (CNN) model is trained to recognize handwritten digits from the MNIST dataset having 99% accuracy on the test dataset.

Model Architecture

The CNN model architecture is defined as follows:

- Input Layer: 28x28 grayscale images
- Convolutional Layer 1: 32 filters, 3x3 kernel size, ReLU activation
- Max Pooling Layer 1: 2x2 pool size
- Convolutional Layer 2: 64 filters, 3x3 kernel size, ReLU activation
- Max Pooling Layer 2: 2x2 pool size
- Flatten Layer: Convert 2D feature maps to 1D
- Dropout Layer: Dropout rate of 0.5 for regularization
- Output Layer: 10 units (corresponding to digits 0-9), softmax activation

Task of model	Number of Layers	Type of Layers	Output Shape	Parameters
Handwritten digit classification	10	Input – input_1	(28, 28, 1)	0
		Convolutional – conv2d	(26, 26, 32)	320
		Activation – conv2d_relu	(26, 26, 32)	0
		Pooling – max_pooling2d	(13, 13, 32)	0
		Convolutional – conv2d_1	(11, 11, 64)	18496
		Activation – conv2d_1_relu	(11, 11, 64)	0
		Pooling – max_pooling2d_1	(5, 5, 64)	0
		Flatten – flatten	(1600, 1)	0
		Dense (Output)–dense	(10, 1)	16010
		Softmax – dense_softmax	(10, 1)	0

2. Changes made to make keras2c generated files synthesizable and a brief description of the change made.

1. Decomposed the k2c_tensor structure to separate variables for each structure defined

Example :

Original

```
k2c_tensor dense_bias = {&dense_bias_array[0],1,10,{10, 1, 1, 1, 1}};
```

To

```
size_t dense_bias_ndim = 1;
```

```
size_t dense_bias_numel = 10;
```

```
size_t dense_bias_shape[5] = {10, 1, 1, 1, 1};
```

```
float dense_bias_array[10] = {...};
```

And so the function call and definition

2. Changed the memcpy and memset to for loop
3. Hoisted all the required variables as global

3. Changes made to generate HLS4ML report if a pragma is removed in this process. For each of the removed pragma, a valid argument must be mentioned.

i) Changed the strategy of layers 'conv2d_1' and 'dense' from default '**latency**' to '**resource**' upon encountering the following errors resulting in pre-synthesis failure :

1. Loop unrolling error citing large runtime and excessive memory usage due to increase in code size.
2. Bitwidth of reshaped elements exceeds the maximum bitwidth (65536 bits) for array.

Upon consulting the documentation, we found the errors to be triggered by Vivado enforced fixed limits by the compiler beyond which maximally unrolled (=parallel) compilation will fail. For the lowest possible latency, each layer should have a maximum number of trainable parameters of 4096. Whereas, in case of our model the above mentioned two layers have trainable parameters 18496 and 16010 respectively, thereby exceeding the limit.

ii) Commented line 167 in file '*build_prj.tcl*' after compilation :

```
config_schedule -enable_dsp_full_reg=false
```

causing error

```
'config_schedule': Unknown option '-enable_dsp_full_reg=false'.
```

FORMAT

```
config_schedule [OPTIONS]
```

```
-effort ( high | medium | low )
```

```
-relax_ii_for_timing
```

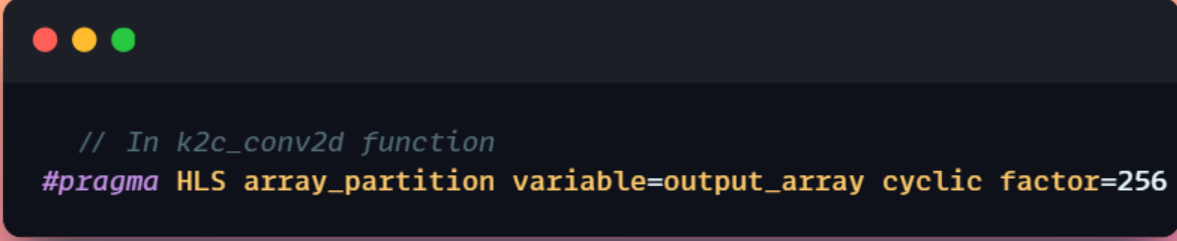
```
-verbose
```

iii) Experimented with various '**reuse factor**' of dsps (multipliers) in order to find a good tradeoff between area and latency.

4. In a markdown cell of jupyter notebook mention all the issues that are faced(dependencies and versions) and solutions to resolve.

5. Optimizations: For each optimization applied (pragma), justify why it has been used.

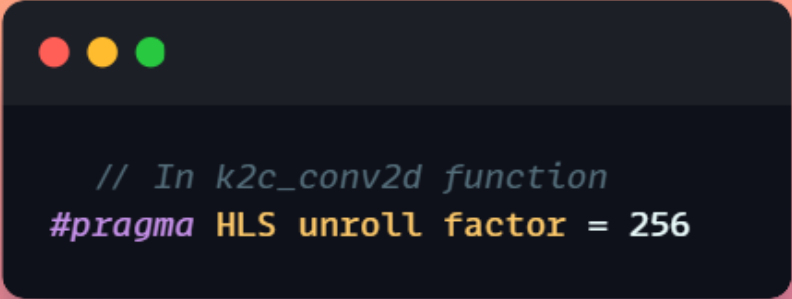
Array Partition



```
// In k2c_conv2d function  
#pragma HLS array_partition variable=output_array cyclic factor=256
```

Partitioned the output_array in k2c_conv2d function for parallel access in unrolled loop

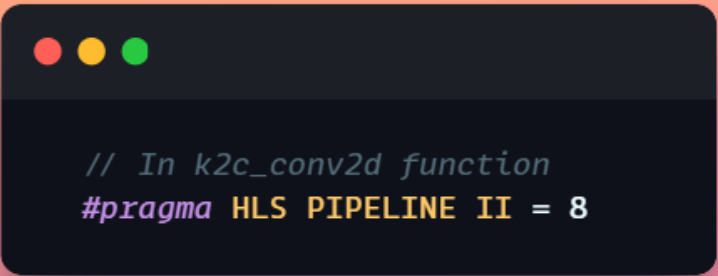
Array Unroll



```
// In k2c_conv2d function  
#pragma HLS unroll factor = 256
```

Unrolled loop for 256 times, for better performance with trade off of hardware resources

Pipeline



```
// In k2c_conv2d function  
#pragma HLS PIPELINE II = 8
```

For parallel access and better performance

Similar Optimization done in other functions as well

Shifted all the required k2c functions to main c file

Changed all the variable loop bounds to the maximum argument of all its calls

Reduced the array access by storing the repeated call of array variables to constant variables

Dead Code elimination

- a. Checked various dataflows and removed the parts which were never called.
- b. Removed unused conditions, loops and variables, arguments

Code Motion

- a. Shifted the heavy computation occurring repeated times, outside the loop
- b. Shifted the constant array access outside the loop

Applied Constant folding

Inlined the small functions

- a. k2c_affine_matmul
- b. k2c_softmax_func
- c. k2c_relu_func
- d. k2c_bias_add
- e. k2c_flatten

6. Results:

- Latency and area overhead table for **Baseline (Unoptimized)**.

Baseline

Summary

Latency		Interval	
min	max	min	max
?	?	?	?

BRAM	DSP48E	FF	LUT
142	33	4861	8000

• **Latency and area overhead table for Optimized**

Summary

Latency		Interval	
min	max	min	max
402173	100631076	402173	100631076

BRAM	DSP48E	FF	LUT
340	110	37879	109211

k2c_conv2d

Latency		Interval	
min	max	min	max
389724	100481075	389792	100481143

BRAM	DSP48E	FF	LUT
257	10	4880	26844

k2c_maxpool2d

Latency		Interval	
min	max	min	max
16229	16229	16229	16229

BRAM	DSP48E	FF	LUT
12	0	360	806

k2c_dense

Latency		Interval	
min	max	min	max
128126	128126	128126	128126

BRAM	DSP48E	FF	LUT
29	57	6116	8789

RTL	Status	Latency		
		min	avg	max
Verilog	Pass	402173	4051626	100631076

• HLS4ML generated Latency and area overhead table

[ReuseFactor 16 for both conv2d_1 & dense]

C synthesis report

Design	LUT	FF	DSP	BRAM	Latency (min/max)	Clock Period
'myproject' xcvu13p-fl ga2577-2- e	194180	195339	2393	1185	3561/3562	4.215

C/RTL cosimulation report

RTL	Status	Latency		
		min	avg	max
Verilog	Pass	2613	2613	2613

[ReuseFactor 6 for conv2d_1 & 4 for dense]

C synthesis report

Design	LUT	FF	DSP	BRAM	Latency (min/max)	Clock Period
'myproject' xcvu13p-fl ga2577-2- e	271355	253958	7313	3372	2038/2039	4.366

• Finally, a comparison report of both Optimized and HLS4ML generated reports.

	HLS4ML	Optimized	Baseline
Latency Min	3561	402173	?
Latency Max	3562	100631076	?
Interval Min	3561	402173	?
Interval Max	3562	100631076	?
BRAM	1185	340	142
DSP48E	2393	110	33
LUT	194180	109211	8000
FF	195339	37879	4861