CMC Infrastructure for Supporting Cloud and Edge Computing Research

Lowering barriers to technology adoption

YASSINE HARIRI
CMC MICROSYSTEMS

LAB

Shorten the development cycle

Access platform-based microsystems design and prototyping environments

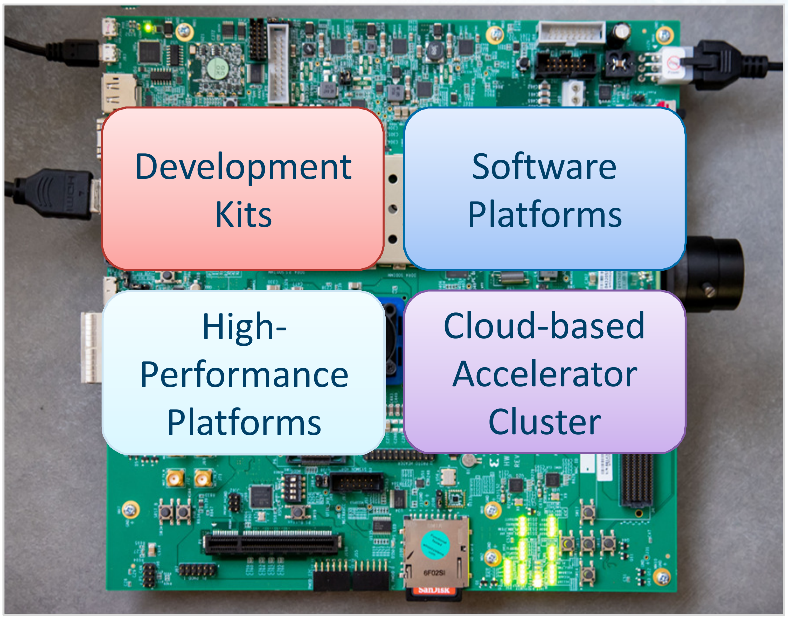
> Systems

> Equipment Rental, Test Fixtures

> Services for emerging processes and products

> Contract R&D

And more: training, webinars, events, CMC engineer support



LAB

www.cmc.ca/Lab-Development-Systems

Cloud

> FPGA/GPU Cluster

Edge

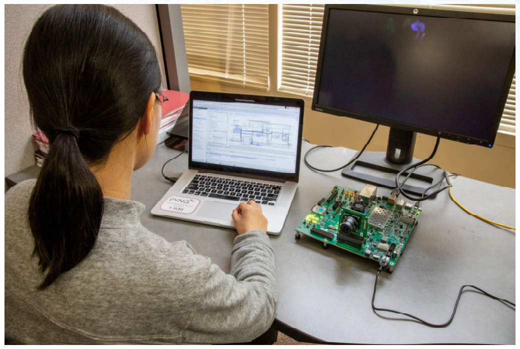
> Xilinx ZCU102 Zynq Ultrascale+ MPSoC Evaluation Kit

RISC-V

> RISC-V Processor Design and Prototyping

Development Kit Rental

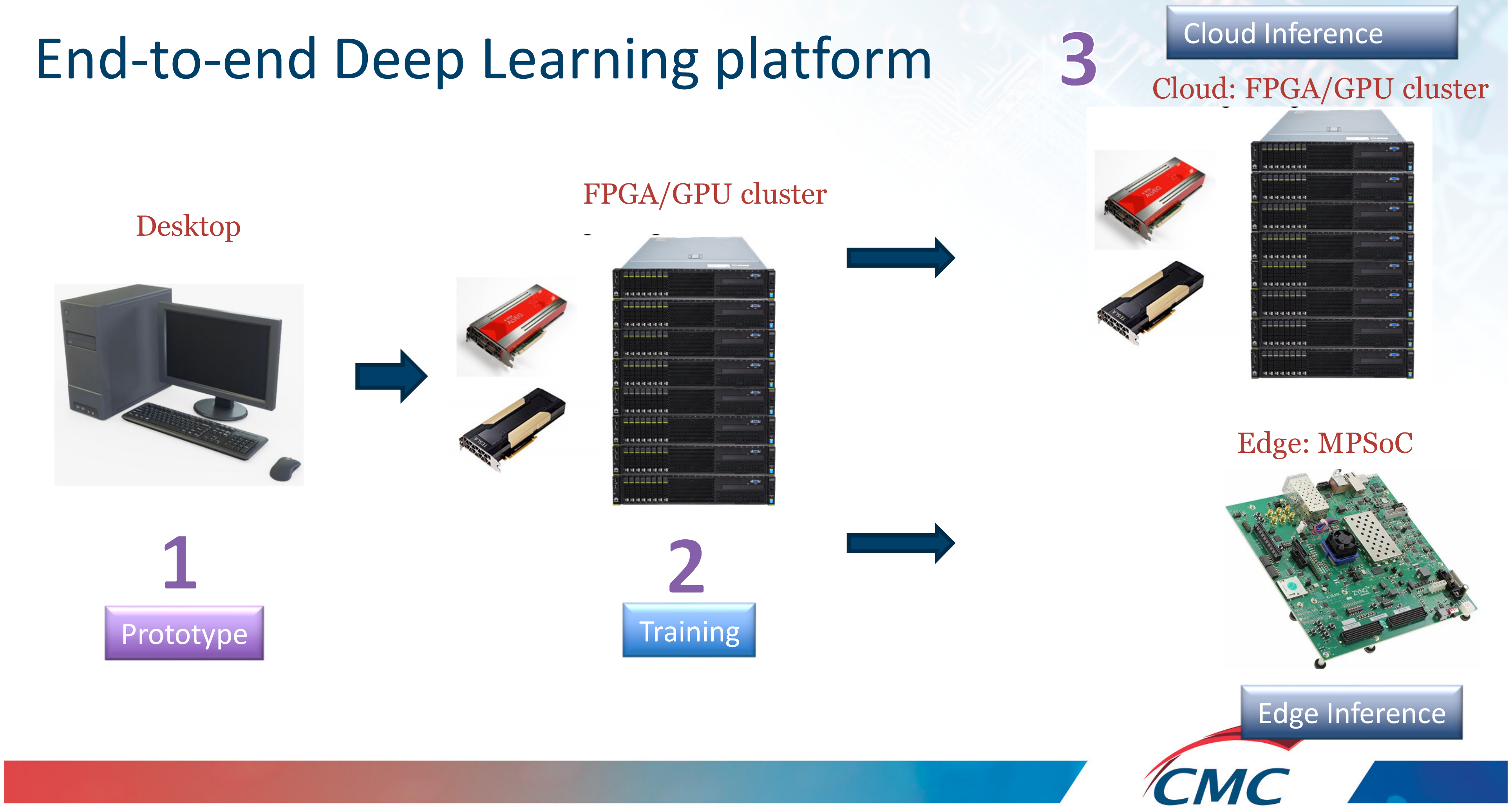
> Xilinx ZCU102 Zynq Ultrascale+ MPSoC Evaluation Kit



Machine Learning Platform An AI solution for monitoring and interpreting aerial surveillance video and imaging… a project supported by National Defence, Innovation for Defence Excellence and Security (IDEaS) program.

CMC

End-to-end Deep Learning platform



CMC

CMC Cloud FPGA/GPU Cluster

- CPUs, GPUs and FPGAs in pre-validated cluster to scale heterogenous computing workloads
  - Machine learning training and inference (e.g. CNN for object detection, speech recognition)
  - Video Processing / Transcoding, Financial Computing, Database analytics, Networking
  - Quantum chemistry, molecular dynamics, climate and weather, Genomics
  - RISC-V Accelerators in Open Source Cloud Computing

Cluster HW

FPGA/GPU cluster Specifications

Cluster Configuration

Environment	Description	# Nodes
Accel - Cerebro	2 Alveo FPGA U200	3
Accel - Genisys	2 V100 GPUs	3
Accel - Synergy	1 Alveo FPGA U200 1 V100 GPU	2

1 Node Specifications

Dual 12 core 3.0 GHz CPU
192 GB RAM
300 GB local storage
100 Gb EDR node interconnect
10 GbE storage network

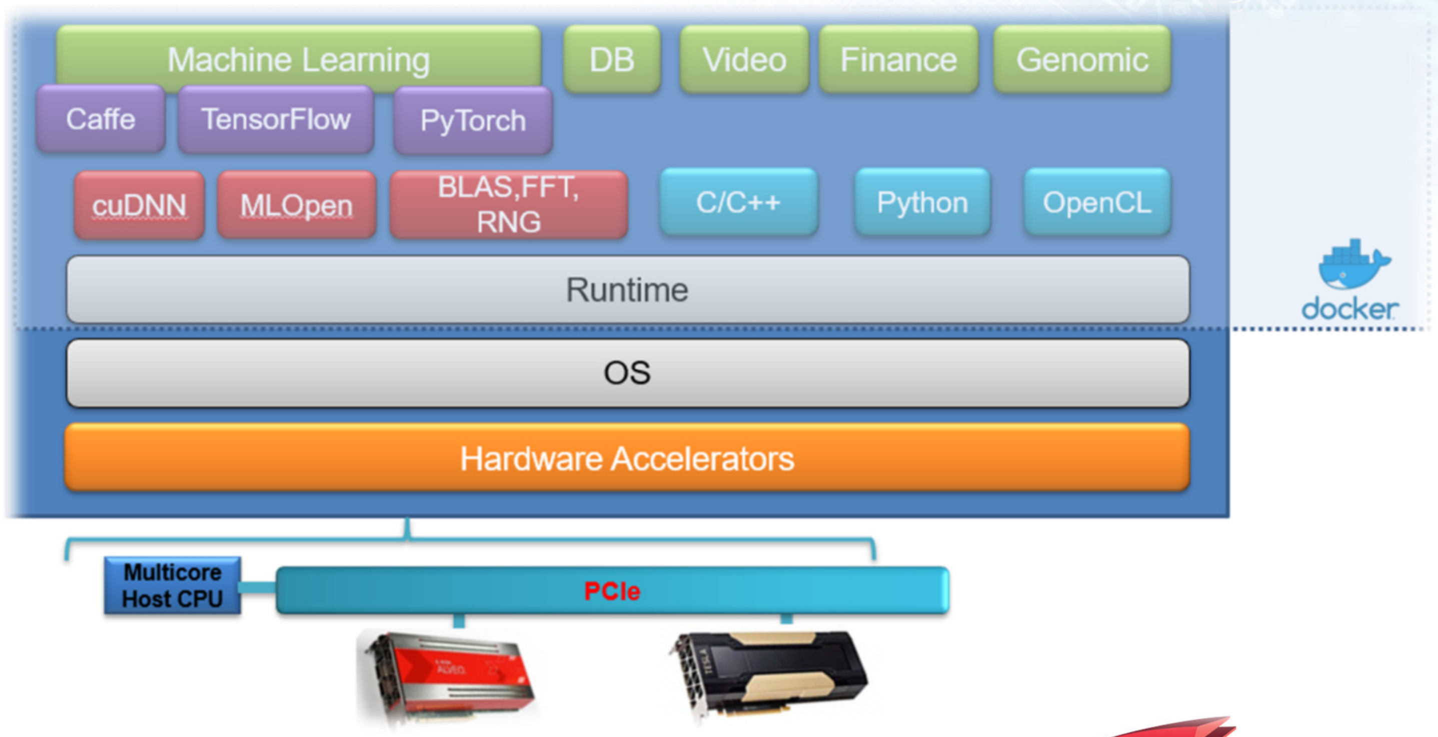
Software stack for the FPGA/GPU cluster

Applications

ML Framework

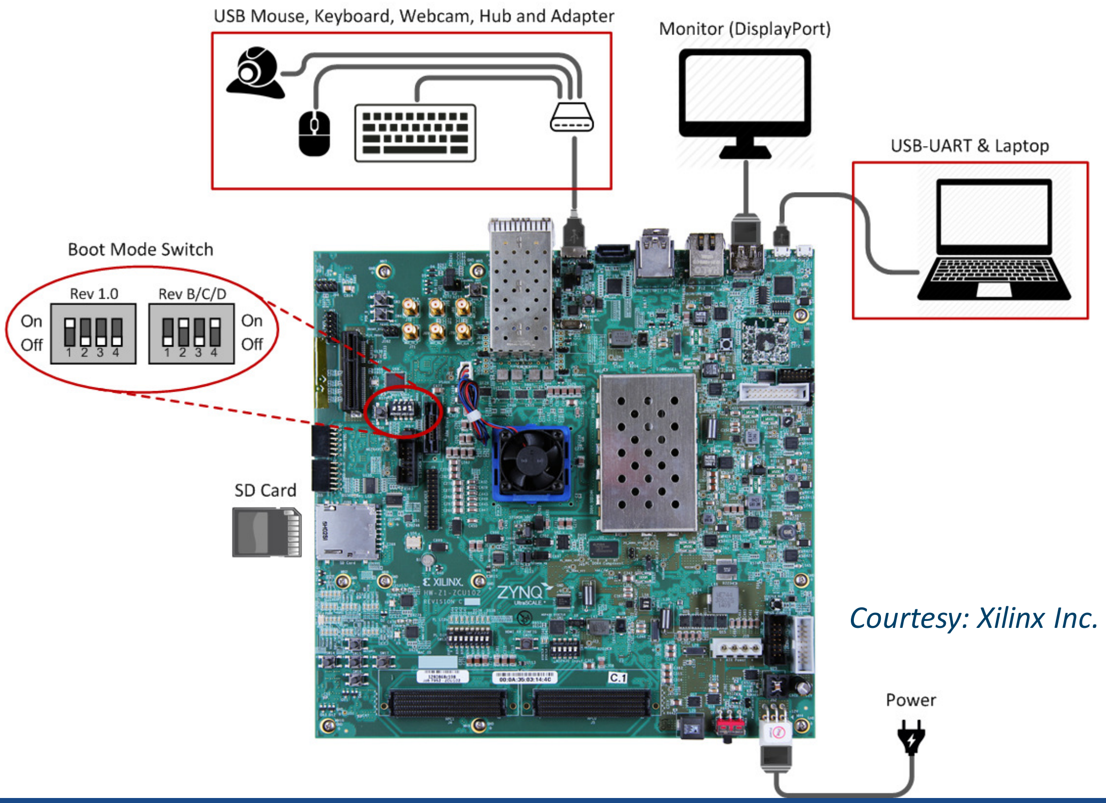
Middleware,
Tools and Libraries

Hardware



Edge Platform: Xilinx ZCU102

- Xilinx Zynq Ultrascale+ MPSoC (ZU9EG)
  - Quad-core ARM A53
  - Dual-core ARM R5
  - ARM GPU
  - 16nm FinFET+ programmable logic
- 4GB 64-bit DDR4 (processor)
- 512MB 16-bit DDR4 (FPGA)
- 2x FMC-HPC connectors
- HDMI video input and output
- DisplayPort video output
- SD Card
- Push buttons, DIP switches, LEDs
- USB UART
- Ethernet



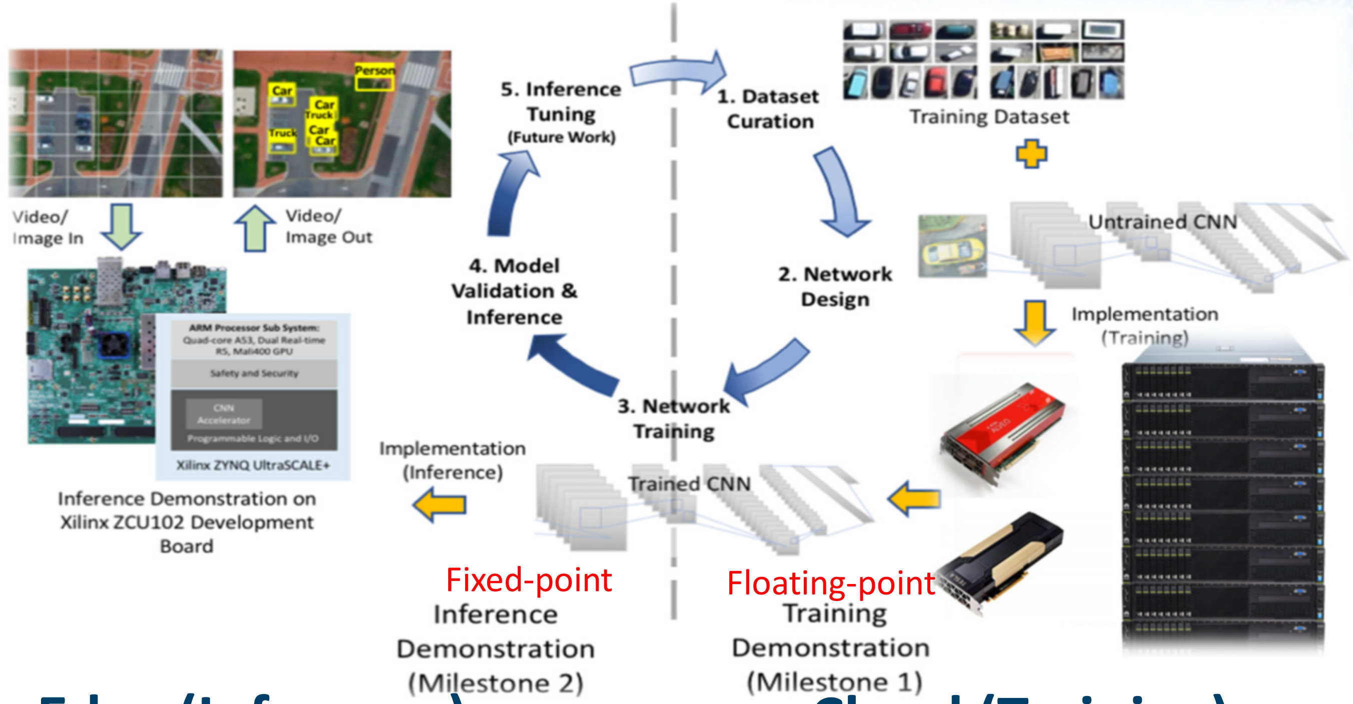
Courtesy: Xilinx Inc.

End-to-end Deep Learning platform

Use case

Innovation for Defence Excellence and Security (IDEaS)

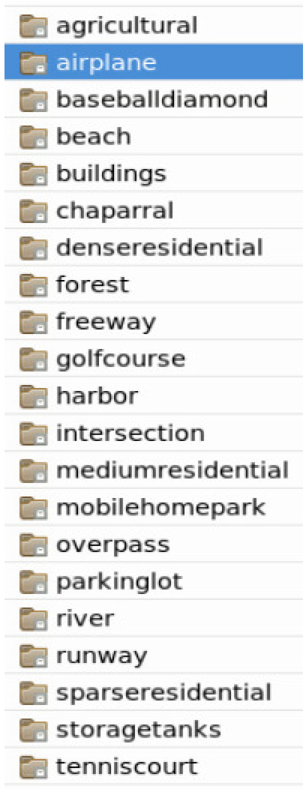
Object Detection, Classification and Tracking Using Heterogeneous Computing Architectures


Car
Car
Truck
Truck
Car
Car
Person
Video/ Image In
Video/ Image Out
ARM Processor Sub System: Quad-core A53, Dual Real-time R5, Mali400 GPU
Safety and Security
CNN Accelerator
Programmable Logic and I/O
Xilinx ZYNQ UltraSCALE+
Inference Demonstration on Xilinx ZCU102 Development Board
5. Inference Tuning (Future Work)
1. Dataset Curation
Training Dataset
Untrained CNN
2. Network Design
Implementation (Training)
4. Model Validation & Inference
3. Network Training
Implementation (Inference)
Trained CNN
Fixed-point Inference Demonstration (Milestone 2)
Floating-point Training Demonstration (Milestone 1)

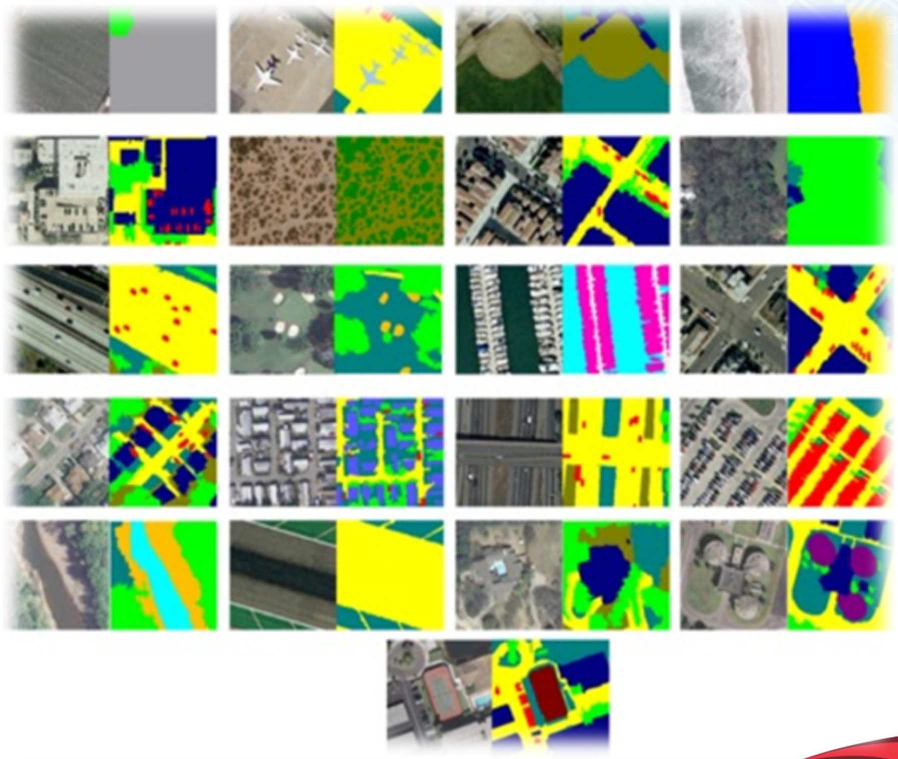
Edge (Inference) **Cloud (Training)**

Phase I: Training Flow

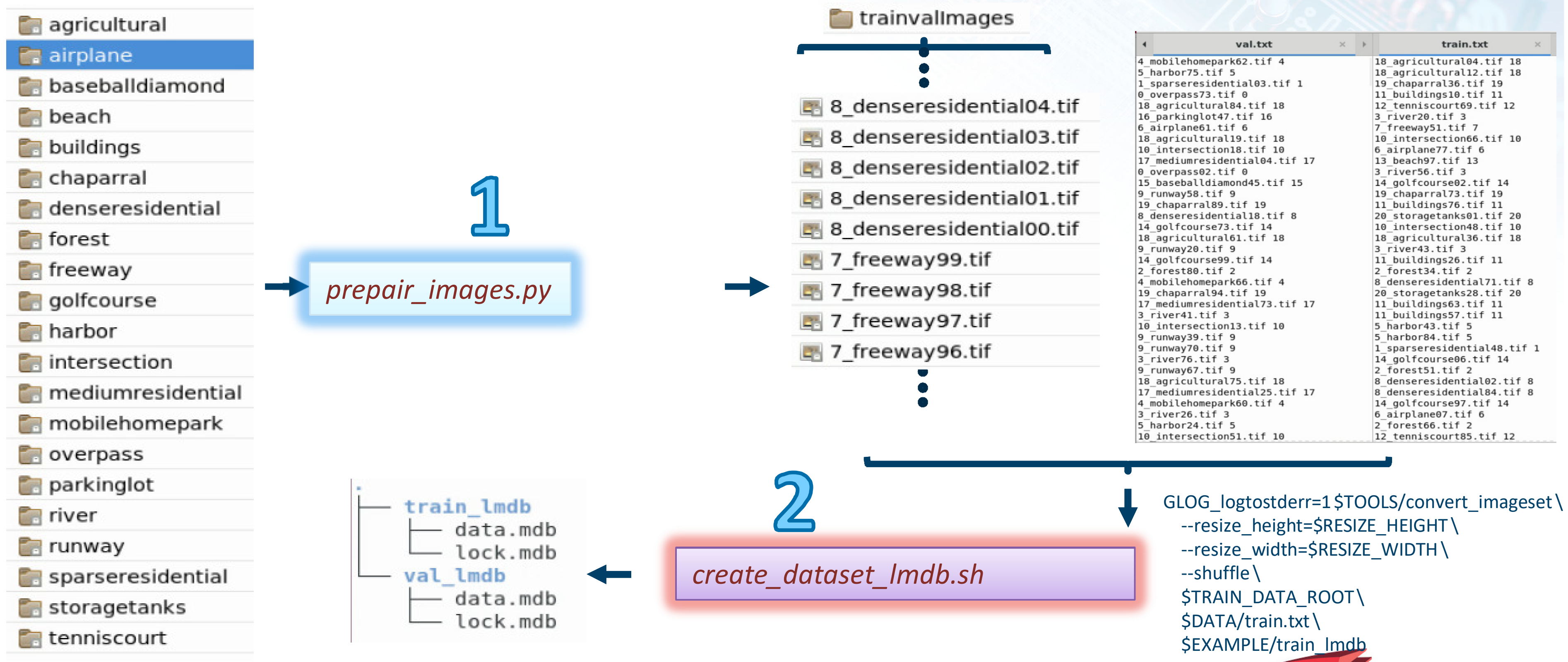
DLRSD dataset


agricultural
airplane
baseballdiamond
beach
buildings
chaparral
denseresidential
forest
freeway
golfcourse
harbor
intersection
mediumresidential
mobilehomepark
overpass
parkinglot
river
runway
sparseresidential
storagetanks
tenniscourt

2100 images 256x256 pixels, 21 class labels



Step 1 - Data preparation



```
agricultural
airplane
baseballdiamond
beach
buildings
chaparral
denseresidential
forest
freeway
golfcourse
harbor
intersection
mediumresidential
mobilehomepark
overpass
parkinglot
river
runway
sparseresidential
storagetanks
tenniscourt
```

```
trainvalImages
8_denseresidential04.tif
8_denseresidential03.tif
8_denseresidential02.tif
8_denseresidential01.tif
8_denseresidential00.tif
7_freeway99.tif
7_freeway98.tif
7_freeway97.tif
7_freeway96.tif
```

val.txt

```
4_mobilehomepark62.tif 4
5_harbor75.tif 5
1_sparseresidential03.tif 1
0_overpass73.tif 0
18_agricultural84.tif 18
16_parkinglot47.tif 16
6_airplane61.tif 6
18_agricultural19.tif 18
10_intersection18.tif 10
17_mediumresidential04.tif 17
0_overpass02.tif 0
15_baseballdiamond45.tif 15
9_runway58.tif 9
19_chaparral89.tif 19
8_denseresidential18.tif 8
14_golfcourse73.tif 14
18_agricultural61.tif 18
9_runway20.tif 9
14_golfcourse99.tif 14
2_forest80.tif 2
4_mobilehomepark66.tif 4
19_chaparral94.tif 19
17_mediumresidential73.tif 17
3_river41.tif 3
10_intersection13.tif 10
9_runway39.tif 9
9_runway70.tif 9
3_river76.tif 3
9_runway67.tif 9
18_agricultural75.tif 18
17_mediumresidential25.tif 17
4_mobilehomepark60.tif 4
3_river26.tif 3
5_harbor24.tif 5
10_intersection51.tif 10
```

train.txt

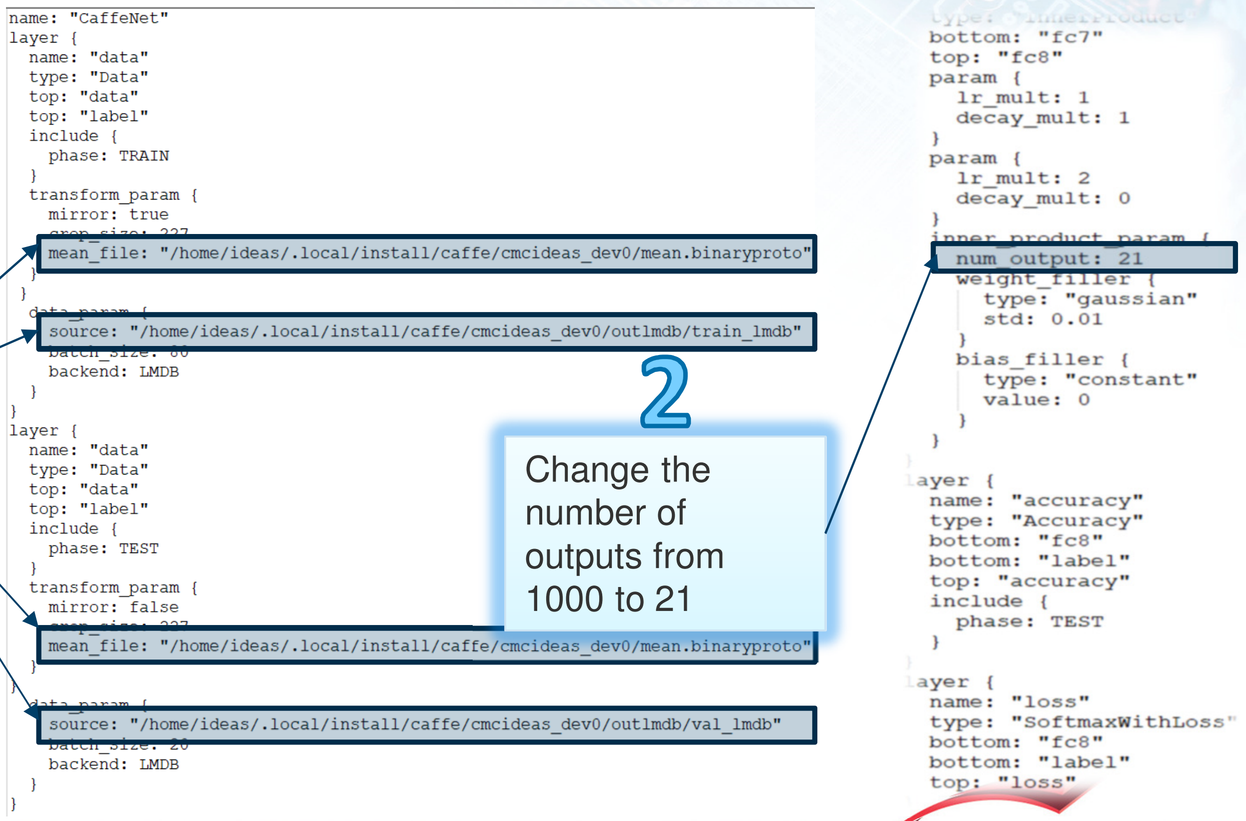
```
18_agricultural04.tif 18
18_agricultural12.tif 18
19_chaparral36.tif 19
11_buildings10.tif 11
12_tenniscourt69.tif 12
3_river20.tif 3
7_freeway51.tif 7
10_intersection66.tif 10
6_airplane77.tif 6
13_beach97.tif 13
3_river56.tif 3
14_golfcourse02.tif 14
19_chaparral73.tif 19
11_buildings76.tif 11
20_storagetanks01.tif 20
10_intersection48.tif 10
18_agricultural36.tif 18
3_river43.tif 3
11_buildings26.tif 11
2_forest34.tif 2
8_denseresidential71.tif 8
20_storagetanks28.tif 20
11_buildings63.tif 11
11_buildings57.tif 11
5_harbor43.tif 5
5_harbor84.tif 5
1_sparseresidential48.tif 1
14_golfcourse06.tif 14
2_forest51.tif 2
8_denseresidential02.tif 8
8_denseresidential84.tif 8
14_golfcourse97.tif 14
6_airplane07.tif 6
2_forest66.tif 2
12_tenniscourt85.tif 12
```

```
GLOG_logtostderr=1 $TOOLS/convert_imageset \
   --resize_height=$RESIZE_HEIGHT \
   --resize_width=$RESIZE_WIDTH \
   --shuffle \
   $TRAIN_DATA_ROOT \
   $DATA/train.txt \
   $EXAMPLE/train_lmdb
```

```
.
├── train_lmdb
│   ├── data.mdb
│   └── lock.mdb
└── val_lmdb
    ├── data.mdb
    └── lock.mdb
```

Step 2 - Model definition

caffenet_train_val_1.prototxt



```
name: "CaffeNet"
layer {
  name: "data"
  type: "Data"
  top: "data"
  top: "label"
  include {
    phase: TRAIN
  }
  transform_param {
    mirror: true
    crop_size: 227
    mean_file: "/home/ideas/.local/install/caffe/cmcideas_dev0/mean.binaryproto"
  }
 }
  data_param {
    source: "/home/ideas/.local/install/caffe/cmcideas_dev0/outlmdb/train_lmdb"
    batch_size: 80
    backend: LMDB
  }
}
layer {
  name: "data"
  type: "Data"
  top: "data"
  top: "label"
  include {
    phase: TEST
  }
  transform_param {
    mirror: false
    crop_size: 227
    mean_file: "/home/ideas/.local/install/caffe/cmcideas_dev0/mean.binaryproto"
  }
}
  data_param {
    source: "/home/ideas/.local/install/caffe/cmcideas_dev0/outlmdb/val_lmdb"
    batch_size: 20
    backend: LMDB
  }
}
```

```
  type: "InnerProduct"
  bottom: "fc7"
  top: "fc8"
  param {
    lr_mult: 1
    decay_mult: 1
  }
  param {
    lr_mult: 2
    decay_mult: 0
  }
  inner_product_param {
    num_output: 21
    weight_filler {
      type: "gaussian"
      std: 0.01
    }
    bias_filler {
      type: "constant"
      value: 0
    }
  }
}
layer {
  name: "accuracy"
  type: "Accuracy"
  bottom: "fc8"
  bottom: "label"
  top: "accuracy"
  include {
    phase: TEST
  }
}
layer {
  name: "loss"
  type: "SoftmaxWithLoss"
  bottom: "fc8"
  bottom: "label"
  top: "loss"
```

1

Change the path for input data and mean image

2

Change the number of outputs from 1000 to 21

Step 3 - Solver definition

- The solver provide parameters to perform model optimisation and guide the training and testing process.
- The content of *solver_1.prototxt* is as follow:

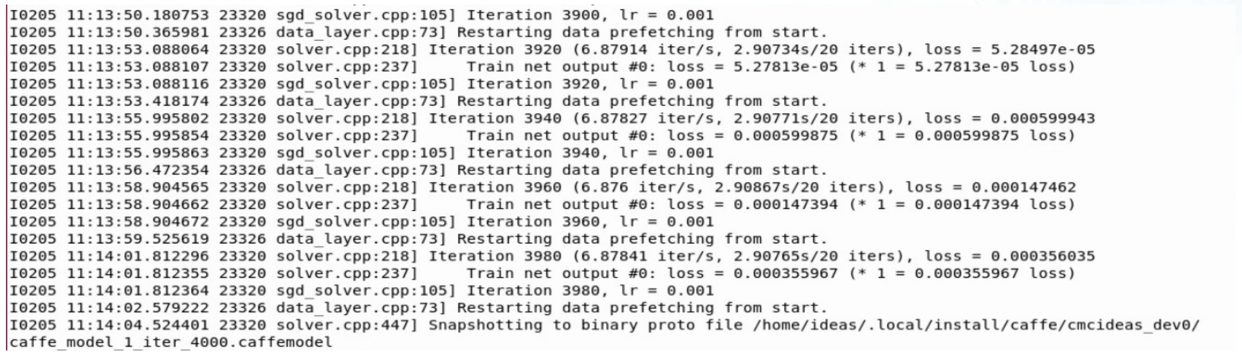
```
net: "/home/ideas/.local/install/caffe/cmcideas_dev0/caffenet_train_val_1.prototxt"
test_iter: 400
test_interval: 500
base_lr: 0.001
lr_policy: "step"
gamma: 0.1
stepsize: 5000
display: 20
max_iter: 10000
momentum: 0.9
weight_decay: 0.0005
snapshot: 2000
snapshot_prefix: "/home/ideas/.local/install/caffe/cmcideas_dev0/caffe_model_1"
solver_mode: GPU
```

Step 4 - Model training

At this step, we are ready to train the model by executing the following CAFFE command from the terminal:

```
>caffe train  solver /home/ideas/.local/install/caffe/cmcideas_dev0/solver_1.prototxt 2>&1 | tee
/home/ideas/.local/install/caffe/cmcideas_dev0/train.log
```

train.log

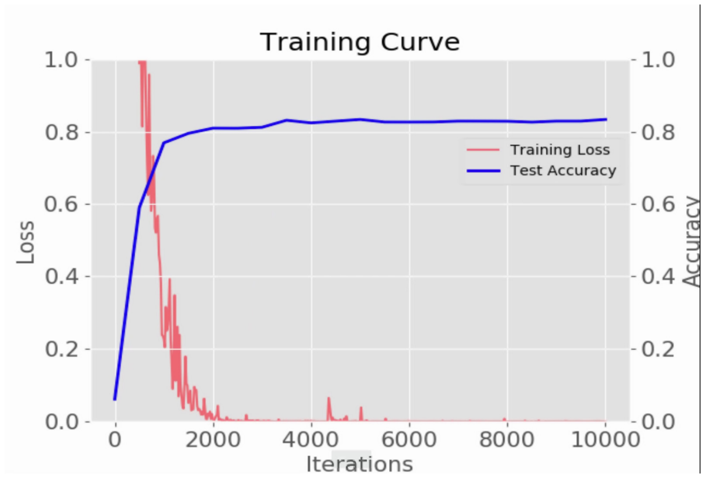


```
I0205 11:13:50.180753 23320 sgd_solver.cpp:105] Iteration 3900, lr = 0.001
I0205 11:13:50.365981 23326 data_layer.cpp:73] Restarting data prefetching from start.
I0205 11:13:53.088064 23320 solver.cpp:218] Iteration 3920 (6.87914 iter/s, 2.90734s/20 iters), loss = 5.28497e-05
I0205 11:13:53.088107 23320 solver.cpp:237]     Train net output #0: loss = 5.27813e-05 (* 1 = 5.27813e-05 loss)
I0205 11:13:53.088116 23320 sgd_solver.cpp:105] Iteration 3920, lr = 0.001
I0205 11:13:53.418174 23326 data_layer.cpp:73] Restarting data prefetching from start.
I0205 11:13:55.995802 23320 solver.cpp:218] Iteration 3940 (6.87827 iter/s, 2.90771s/20 iters), loss = 0.000599943
I0205 11:13:55.995854 23320 solver.cpp:237]     Train net output #0: loss = 0.000599875 (* 1 = 0.000599875 loss)
I0205 11:13:55.995863 23320 sgd_solver.cpp:105] Iteration 3940, lr = 0.001
I0205 11:13:56.472354 23326 data_layer.cpp:73] Restarting data prefetching from start.
I0205 11:13:58.904565 23320 solver.cpp:218] Iteration 3960 (6.876 iter/s, 2.90867s/20 iters), loss = 0.000147462
I0205 11:13:58.904662 23320 solver.cpp:237]     Train net output #0: loss = 0.000147394 (* 1 = 0.000147394 loss)
I0205 11:13:58.904672 23320 sgd_solver.cpp:105] Iteration 3960, lr = 0.001
I0205 11:13:59.525619 23326 data_layer.cpp:73] Restarting data prefetching from start.
I0205 11:14:01.812296 23320 solver.cpp:218] Iteration 3980 (6.87841 iter/s, 2.90765s/20 iters), loss = 0.000356035
I0205 11:14:01.812355 23320 solver.cpp:237]     Train net output #0: loss = 0.000355967 (* 1 = 0.000355967 loss)
I0205 11:14:01.812364 23320 sgd_solver.cpp:105] Iteration 3980, lr = 0.001
I0205 11:14:02.579222 23326 data_layer.cpp:73] Restarting data prefetching from start.
I0205 11:14:04.524401 23320 solver.cpp:447] Snapshotting to binary proto file /home/ideas/.local/install/caffe/cmcideas_dev0/
caffe_model_1_iter_4000.caffemodel
```

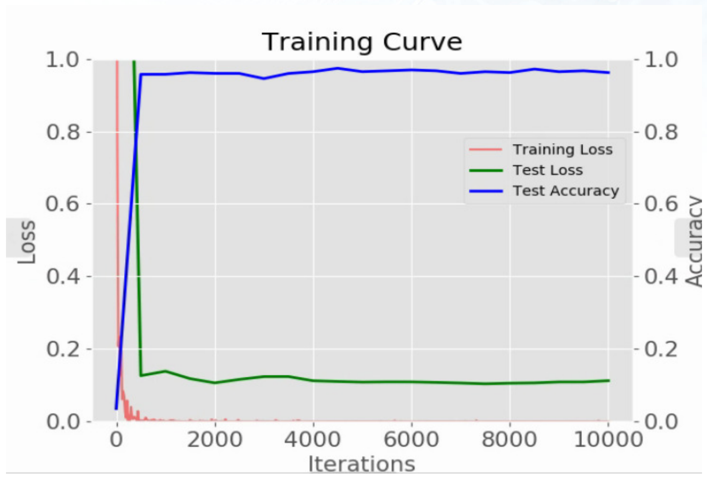
```
>python /home/ideas/.local/install/caffe/cmcideas_dev0/plot_learning_curve.py
/home/ideas/.local/install/caffe/cmcideas_dev0/train.log
/home/ideas/.local/install/caffe/cmcideas_dev0/learning_curve.png
```

Transfer Learning

- Concept: Instead of training the network from scratch, transfer learning trains an already trained model on a different dataset.



- validation accuracy: **~85%,** after **4000** iterations.

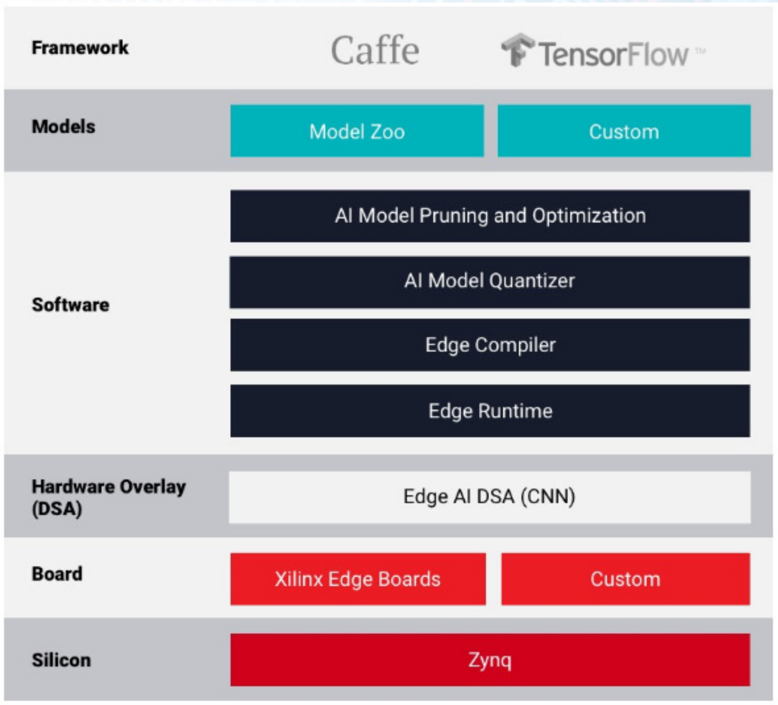


- validation accuracy: **~98%,** after **1000** iterations.

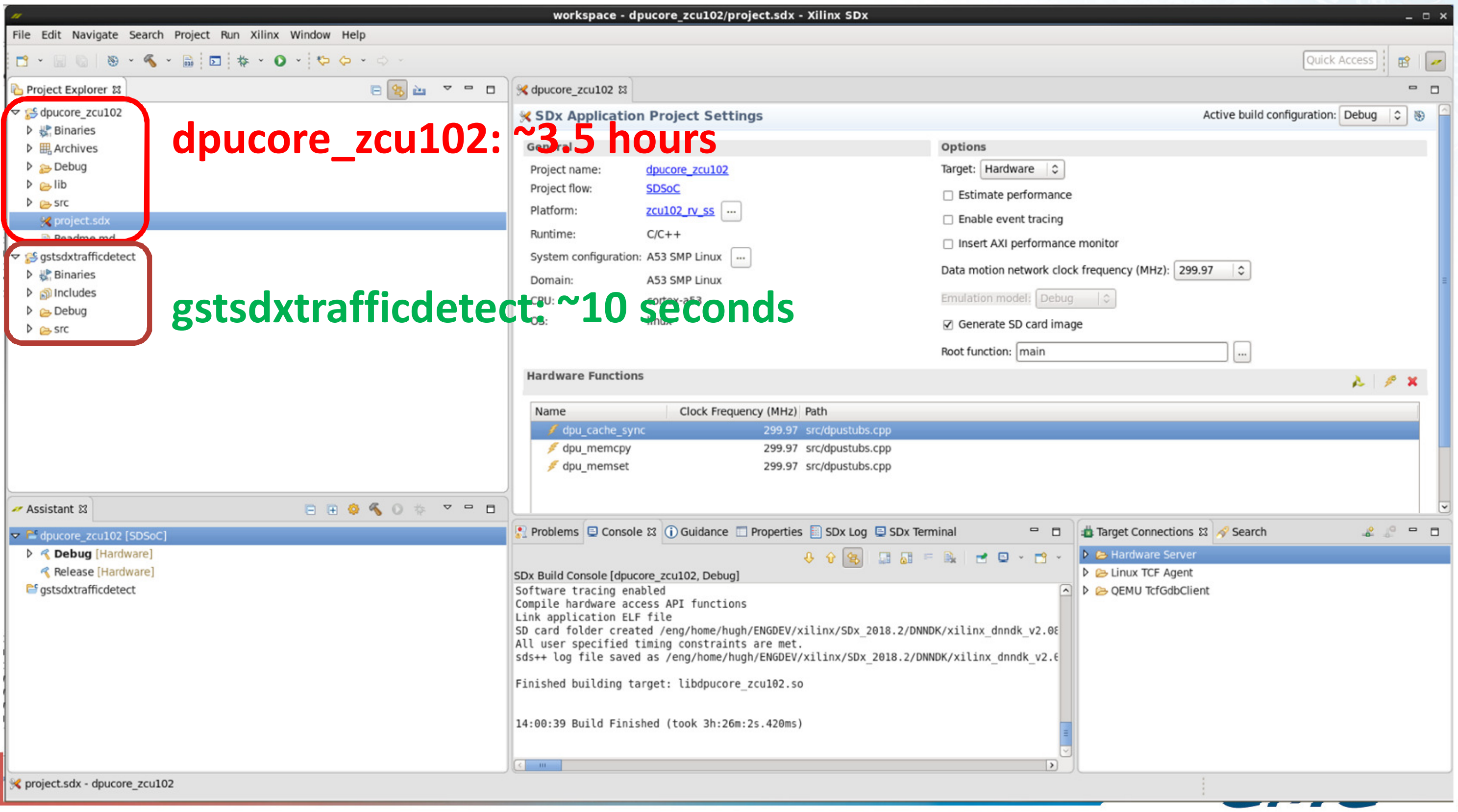
Phase II: Inference Flow

Xilinx DNNDK

- Full-stack SDK for the Deep-learning Processor Unit (DPU)
- Supports CNN quantization, compilation, optimization and runtime support
- Network pruning supported by separate license
- Supports Caffe and TensorFlow
- Freely downloaded from Xilinx (registration required)
- Compatible with existing Xilinx tools/flows (Vivado, SDSoC)
- Supported evaluation boards:
  - ZCU102
  - ZCU104
  - Ultra96



Build Hardware and Application Projects in the SDSoC Development Environment



dpucore_zcu102: ~3.5 hours

gstsdxtrafficdetect: ~10 seconds

SDx Application Project Settings

Active build configuration: Debug

General

Project name:	dpucore_zcu102
Project flow:	SDSoC
Platform:	zcu102_rv_ss
Runtime:	C/C++
System configuration:	A53 SMP Linux
Domain:	A53 SMP Linux
CPU:	cortex-a53
OS:	linux

Options

Target: Hardware

- [ ] Estimate performance
- [ ] Enable event tracing
- [ ] Insert AXI performance monitor

Data motion network clock frequency (MHz): 299.97

Emulation model: Debug

- [x] Generate SD card image

Root function: main

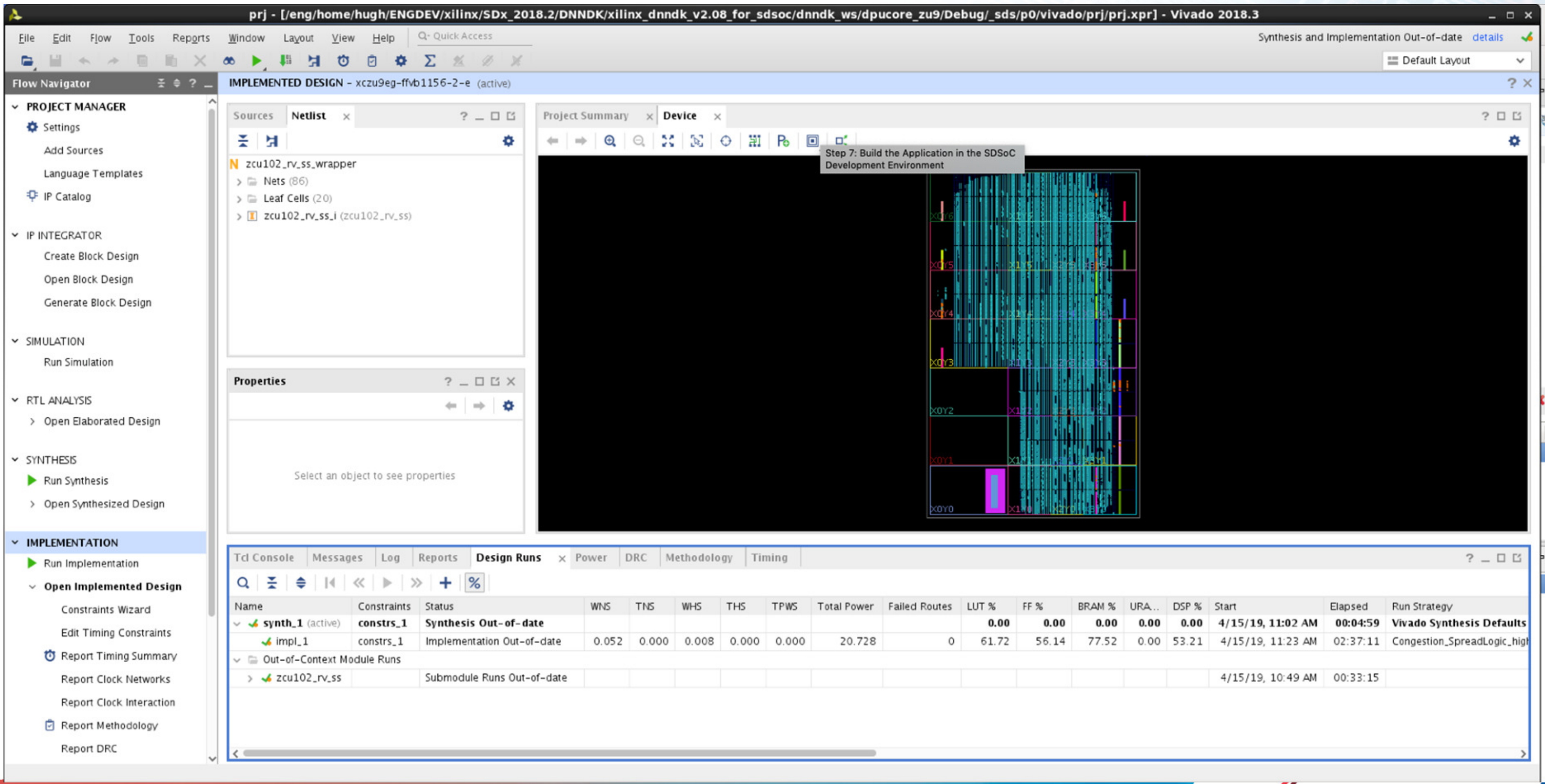
Hardware Functions

Name	Clock Frequency (MHz)	Path
dpu_cache_sync	299.97	src/dpustubs.cpp
dpu_memcpy	299.97	src/dpustubs.cpp
dpu_memset	299.97	src/dpustubs.cpp

SDx Build Console [dpucore_zcu102, Debug]

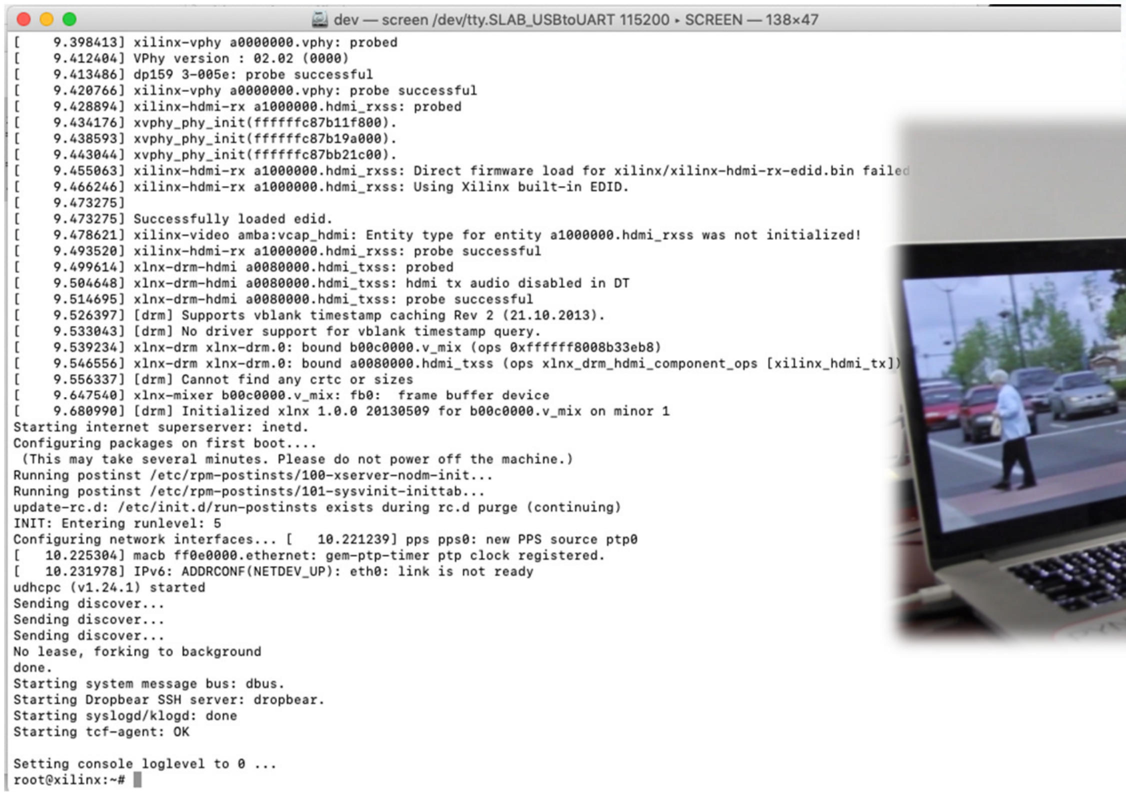
```
Software tracing enabled
Compile hardware access API functions
Link application ELF file
SD card folder created /eng/home/hugh/ENGDEV/xilinx/SDx_2018.2/DNNDK/xilinx_dnndk_v2.08
All user specified timing constraints are met.
sds++ log file saved as /eng/home/hugh/ENGDEV/xilinx/SDx_2018.2/DNNDK/xilinx_dnndk_v2.0

Finished building target: libdpucore_zcu102.so


14:00:39 Build Finished (took 3h:26m:2s.420ms)
```

Implementation results: Xilinx Vivado



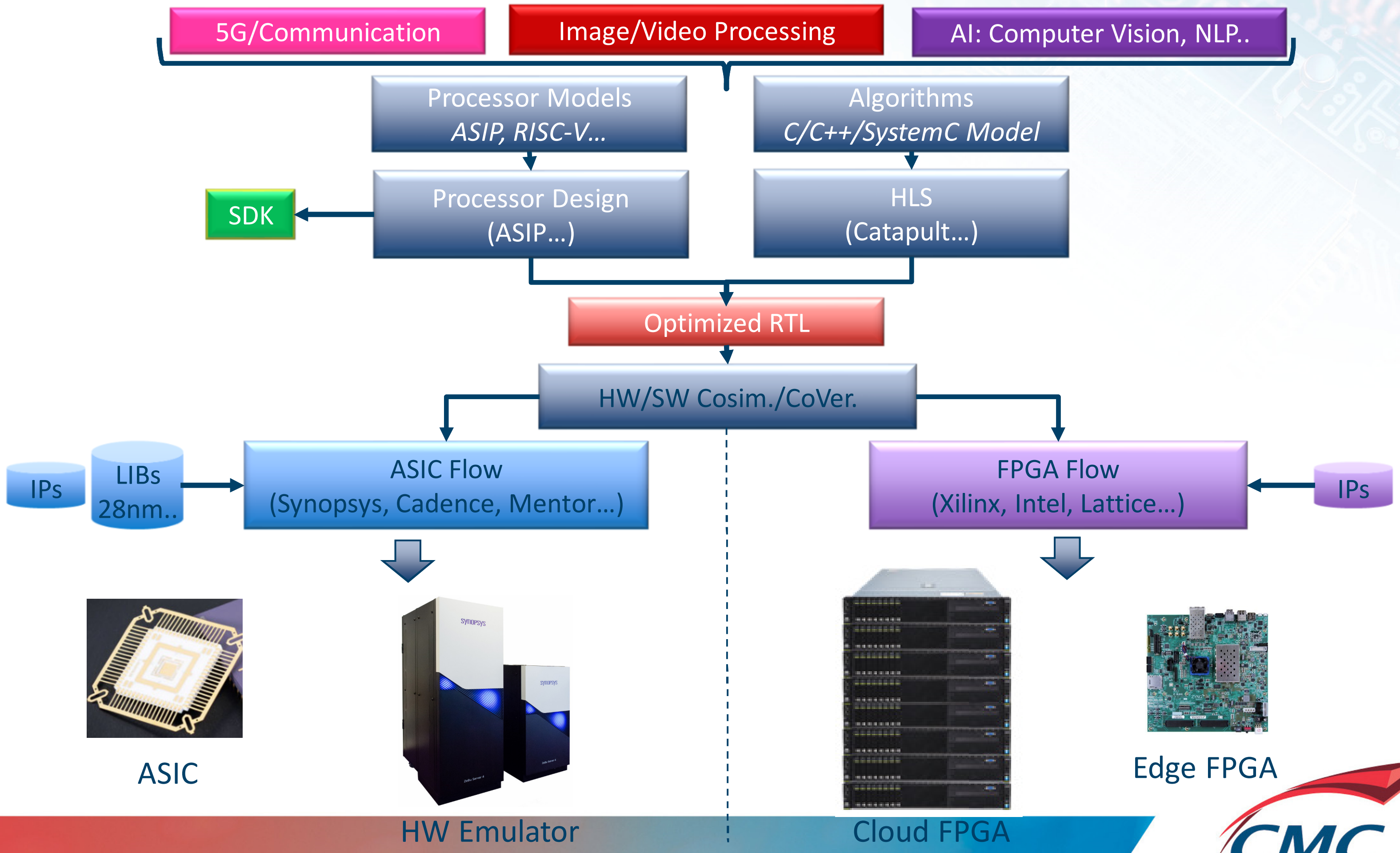
Name	Constraints	Status	WNS	TNS	WHS	THS	TPWS	Total Power	Failed Routes	LUT %	FF %	BRAM %	URA...	DSP %	Start	Elapsed	Run Strategy
synth_1 (active)	**constrs_1**	**Synthesis Out-of-date**								**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**4/15/19, 11:02 AM**	**00:04:59**	**Vivado Synthesis Defaults**
impl_1	constrs_1	Implementation Out-of-date	0.052	0.000	0.008	0.000	0.000	20.728	0	61.72	56.14	77.52	0.00	53.21	4/15/19, 11:23 AM	02:37:11	Congestion_SpreadLogic_high
Out-of-Context Module Runs																	
zcu102_rv_ss		Submodule Runs Out-of-date													4/15/19, 10:49 AM	00:33:15	

Run the application on ZCU102


dev — screen /dev/tty.SLAB_USBtoUART 115200 ▸ SCREEN — 138×47

```
[    9.398413] xilinx-vphy a0000000.vphy: probed
[    9.412404] VPhy version : 02.02 (0000)
[    9.413486] dp159 3-005e: probe successful
[    9.420766] xilinx-vphy a0000000.vphy: probe successful
[    9.428894] xilinx-hdmi-rx a1000000.hdmi_rxss: probed
[    9.434176] xvphy_phy_init(ffffffc87b11f800).
[    9.438593] xvphy_phy_init(ffffffc87b19a000).
[    9.443044] xvphy_phy_init(ffffffc87bb21c00).
[    9.455063] xilinx-hdmi-rx a1000000.hdmi_rxss: Direct firmware load for xilinx/xilinx-hdmi-rx-edid.bin failed
[    9.466246] xilinx-hdmi-rx a1000000.hdmi_rxss: Using Xilinx built-in EDID.
[    9.473275]
[    9.473275] Successfully loaded edid.
[    9.478621] xilinx-video amba:vcap_hdmi: Entity type for entity a1000000.hdmi_rxss was not initialized!
[    9.493520] xilinx-hdmi-rx a1000000.hdmi_rxss: probe successful
[    9.499614] xlnx-drm-hdmi a0080000.hdmi_txss: probed
[    9.504648] xlnx-drm-hdmi a0080000.hdmi_txss: hdmi tx audio disabled in DT
[    9.514695] xlnx-drm-hdmi a0080000.hdmi_txss: probe successful
[    9.526397] [drm] Supports vblank timestamp caching Rev 2 (21.10.2013).
[    9.533043] [drm] No driver support for vblank timestamp query.
[    9.539234] xlnx-drm xlnx-drm.0: bound b00c0000.v_mix (ops 0xffffff8008b33eb8)
[    9.546556] xlnx-drm xlnx-drm.0: bound a0080000.hdmi_txss (ops xlnx_drm_hdmi_component_ops [xilinx_hdmi_tx])
[    9.556337] [drm] Cannot find any crtc or sizes
[    9.647540] xlnx-mixer b00c0000.v_mix: fb0:  frame buffer device
[    9.680990] [drm] Initialized xlnx 1.0.0 20130509 for b00c0000.v_mix on minor 1
Starting internet superserver: inetd.
Configuring packages on first boot....
 (This may take several minutes. Please do not power off the machine.)
Running postinst /etc/rpm-postinsts/100-xserver-nodm-init...
Running postinst /etc/rpm-postinsts/101-sysvinit-inittab...
update-rc.d: /etc/init.d/run-postinsts exists during rc.d purge (continuing)
INIT: Entering runlevel: 5
Configuring network interfaces... [   10.221239] pps pps0: new PPS source ptp0
[   10.225304] macb ff0e0000.ethernet: gem-ptp-timer ptp clock registered.
[   10.231978] IPv6: ADDRCONF(NETDEV_UP): eth0: link is not ready
udhcpc (v1.24.1) started
Sending discover...
Sending discover...
Sending discover...
No lease, forking to background
done.
Starting system message bus: dbus.
Starting Dropbear SSH server: dropbear.
Starting syslogd/klogd: done
Starting tcf-agent: OK

Setting console loglevel to 0 ...
root@xilinx:~# 
```

A Unified Design Flow for Advanced Computing Platforms



CMC

Thank you

Yassine Hariri
Hariri@cmc.ca

www.cmc.ca