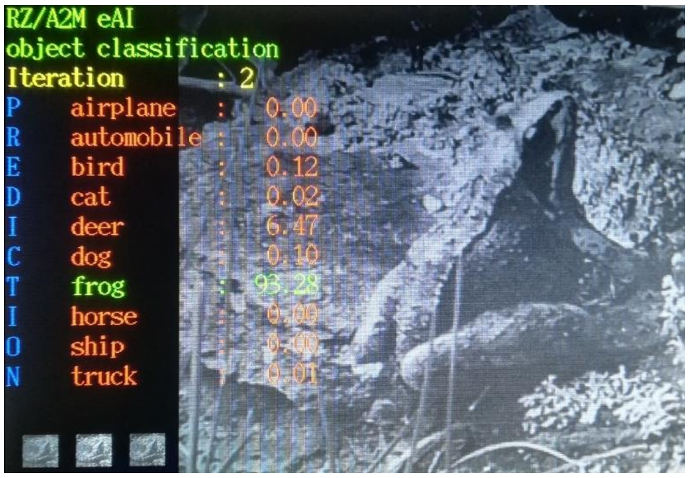
RZ/A2M

RZ/A2M object classification demo - Application Note



Introduction

This document describes the RZ/A2M object classification demo software.

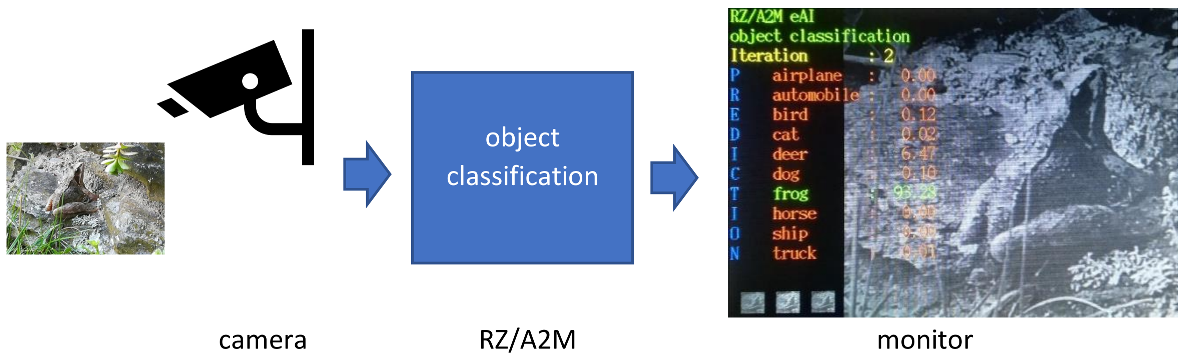
Target Device

RZ/A2M

Contents

1. Overview

The demo classifies objects in front of a camera.
The predicted result is shown on a display (HDMI monitor or RSK display).



The CNN used in this demo has been trained with the CIFAR-10 database and therefore classifies the following 10 different objects only:
'airplane', 'automobile', 'bird', 'cat', 'deer', 'dog', 'frog', 'horse', 'ship' and 'truck'
(Please see https://www.cs.toronto.edu/~kriz/cifar.html for details.)

The software runs on a RZ/A2M development kit.

The display shows the captured image in grayscale.
The user switch SW3 allows selecting the information to be shown at the display.
This is the object prediction result or the turn-around-time of the different processing steps.

The demo is based on the “2D Barcode Sample demo” (rza2m_2d_barcode_sample_freertos_gcc).
Please check it’s document “2D Barcode Application Note”
/RZA2M_object_classification_example/doc/r01an4503ej0101-rza2.pdf
for additional information.
The “2D Barcode Sample demo” has been released within the “RZ/A2M Group RZ/A2M 2D Barcode Package” that is available for download at
https://www.renesas.com/eu/en/products/microcontrollers-microprocessors/rz/rza/rza2m.html#sampleCodes

The routines to detect the 2D Barcode and to encode the Barcode are not required and have been removed. The main modifications have been made in
/RZA2M_object_classification_example/src/renesas/application/r_bcd_main.c

2. Operation Confirmation Conditions

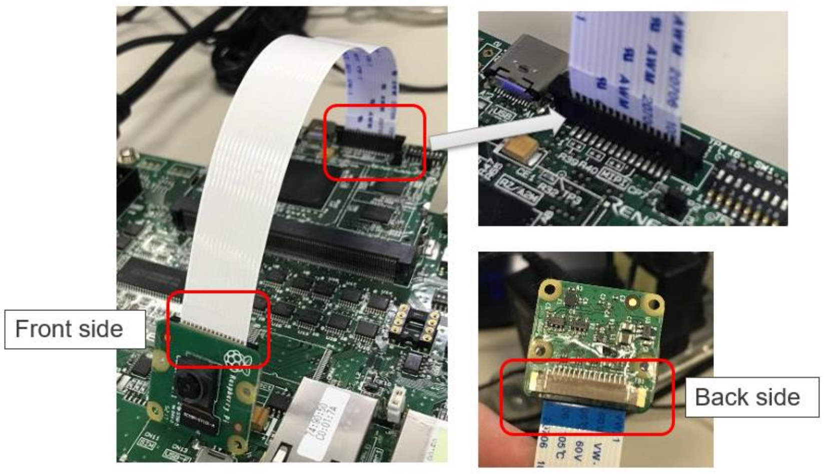
2.1. Evaluation board setup

The board setup is the same as used for the "RZ/A2M Group RZ/A2M Simple Applications Package; Project, C Source (e2 studio project / GNU ARM Embedded)".
Please check one of these projects for
    doc/readme-e.txt
Especially the setting for DIP switches and jumpers need to be followed.
For your convenience, these are given in chapter 2.2 as well.

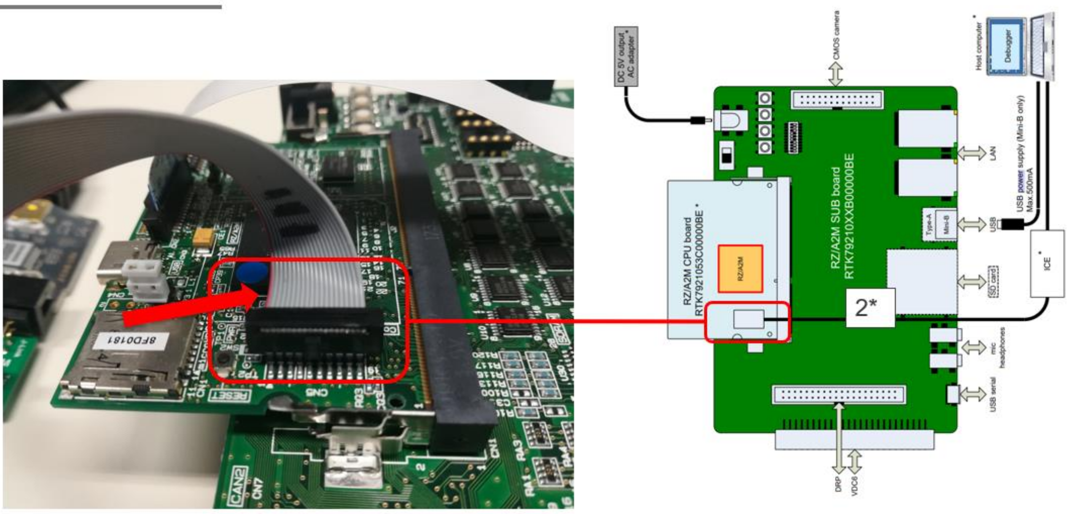
Additional information can be found in the "2D Barcode Application Note"
    /RZA2M_object_classification_example/doc/r01an4503ej0101-rza2.pdf
.

2.2. Evaluation board setup (quick start)

MIPI CAMERA CONNECTION

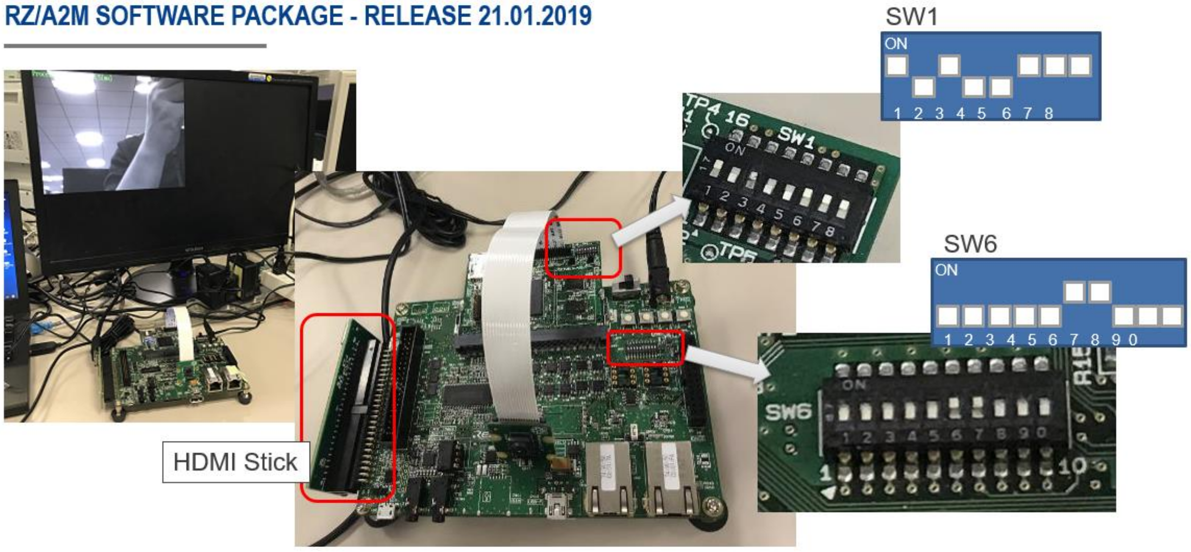


J_LINK CONNECTION


DC 5V output AC adapter *
CMOS camera
Host computer *
Debugger
USB power supply (Mini-B only) Max 500mA
ICE *
RZ/A2M CPU board RTK7921053C00000BE *
RZ/A2M
RZ/A2M SUB board RTK79210XXB00000BE
LAN
Type-A
Mini-B
USB
SD card
2*
mic
headphones
USB serial
DRP
VDC6

DIP SWITCHES AND JUMPERS

RZ/A2M SOFTWARE PACKAGE - RELEASE 21.01.2019



1. Software

1.1. Folder Structure

```
RZA2M_number_plate_detection
+---bootloader
+---doc
+---generate
|   +---compiler
|   +---configuration
|   +---drivers
|   +---os_abstraction
|   +---sc_drivers
|   |   +---r_cbuffer
|   |   +---r_ceu
|   |   +---r_drp
|   |   |   +---doc
|   |   |   +---drp_lib
|   |   +---r_mipi
|   |   +---r_ostm
|   |   +---r_riic
|   |   +---r_rvapi
|   |   +---r_scifa
|   |   \---r_vdc
|   \---system
\---src
    +---freertos
    +---renesas
    |   +---application
    |   |   +---common
    |   |   |   +---camera
    |   |   |   +---perform
    |   |   |   +---port_settings
    |   |   |   \---render
    |   |   \---inc
    |   |       +---camera
    |   |       \---lcd
    \---Translator
    \---user_prog
```

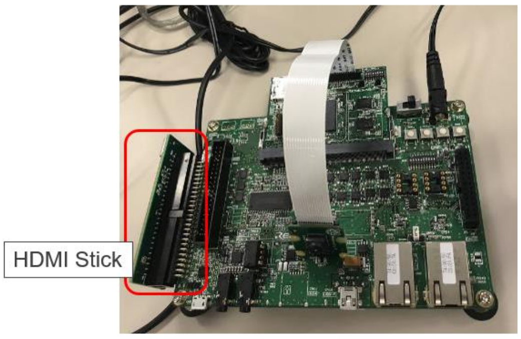
FreeRTOS is open source software distributed under the MIT license.
Regarding the MIT license,please refer to
https://opensource.org/licenses/mit-license.php
FreeRTOS is a real-time operation system kernel for embedded microcomputers. In this sample program, Kernel v10.0.0 is used.

1.2. Display type definition

The RZ/A2M development kit offers different options to connect a display.
To change the display type, please open header file
/ RZA2M_object_classification_example/src/renesas/application/inc/lcd_panel.h
and map LCD_PANEL with one of the pre-defined options (e.g. LCD_PANEL_RSK or LCD_PANEL_DVI).

LCD_PANEL_RSK is the label for a Renesas touch display that can be connected to the development board directly.

LCD_PANEL_DVI is the label for HDMI displays that can be connected via the HDMI interface stick that needs to be connected to the development board.



If LCD_PANEL_DVI is selected, please check header file
/ RZA2M_object_classification_example/src/renesas/application/inc/lcd/pc_monitor.h
and modify the define for SELECT_MONITOR to select the most suitable configuration for the connected monitor.

1.3. DRP libraries

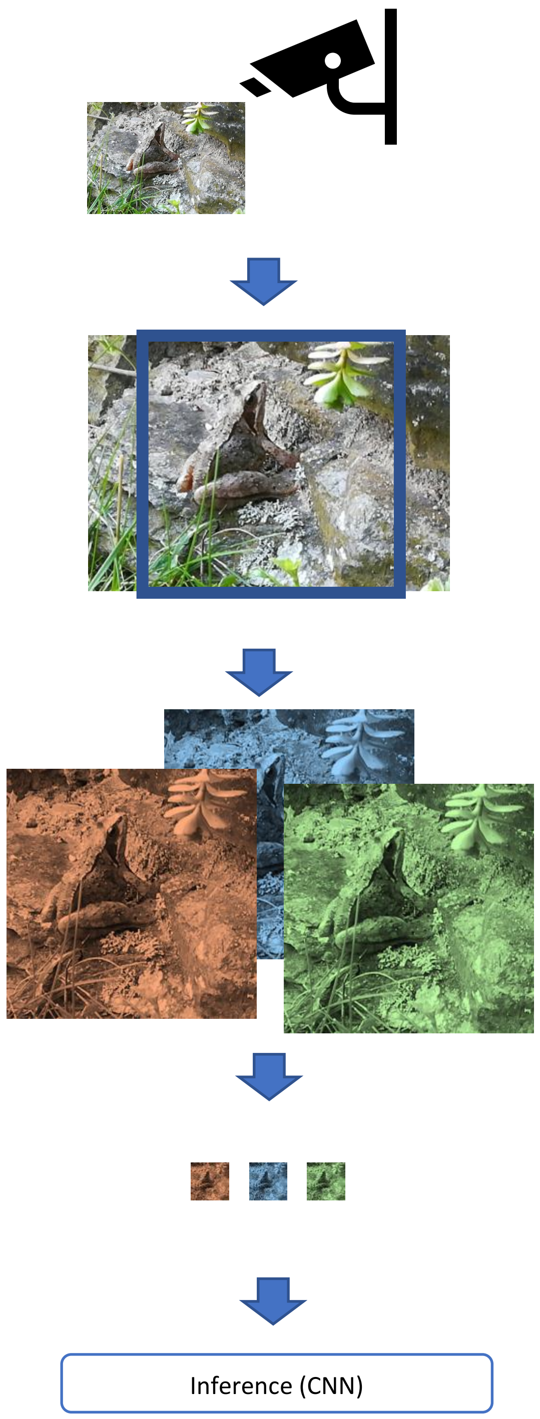
The Software uses a mixture of released DRP libraries and one DRP library (rgb2splitrgb) that has been developed at Renesas Electronics Europe (SoCDD).

The rgb2splitrgb libraries is given in

/RZA2M_object_classification_example/generate/sc_drivers/r_drp/drp_lib/r_drp_rgb2splitrgb

2. Image Pre-Processing

2.1. Pre-Processing before step “inference”



Camera (Raspberry Pi camera V2.1)

Bayer to RGB conversion

(DRP library “bayer2rgb”)

image size 640 x 480 pixel

Split RGB format (RGB888) to R8, G8, B8
and crop centre image of 480 x 480 pixel.

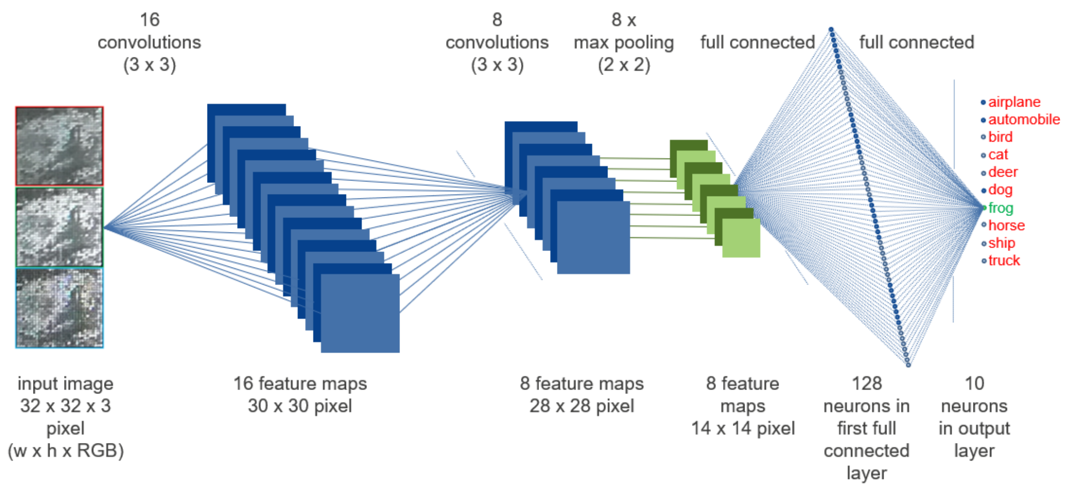
(DRP library “rgb2splitrgb”)

Resize all 3 images to 32 x 32 pixel

(DRP library “resize_bilinear”)

3. Inference with a Convolutional Neural Network (CNN)

CNN architecture



The CNN is described and trained within jupyter notebooks importing the Keras/TensorFlow AI framework. (Library versions: Keras 2.2.4; TensorFlow 1.12.0).

```
model = Sequential()

model.add(Conv2D(16, (3, 3), activation='relu', input_shape=(32, 32, 3)))
model.add(Conv2D(8, (3, 3), activation='relu'))

model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Flatten())

model.add(Dense(128, activation='relu'))
model.add(Dense(10, activation='softmax'))

sgd = SGD(lr=0.01, decay=1e-6, momentum=0.9, nesterov=True)
model.compile(loss='categorical_crossentropy', metrics=['accuracy'], optim
izer=sgd)
```

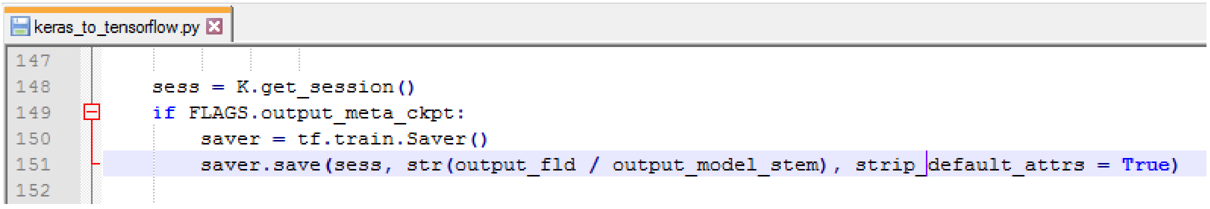
‘Softmax’ has been chosen as activation function for the output layer.
This amplifies the prediction accuracy of the ‘best fit’, but decreases the ability to differentiate between a ‘week’ and ‘strong’ predictions.
(Changing the activation function from ‘softmax’ to ‘relu’ may be evaluated within a release update.)

3.1. NN translation (eAI translator)

Keras models cannot be handled by the Renesas eAI translator directly.
It’s mandatory to port the Keras model to the original TensorFlow format.
This can be done with a slightly modified script from github.

Python script to port Keras to TensorFlow
https://github.com/amir-abdi/keras_to_tensorflow

Modify the keras_to_tensorflow.py script to write out the frozen model with ‘forward compatibility’.

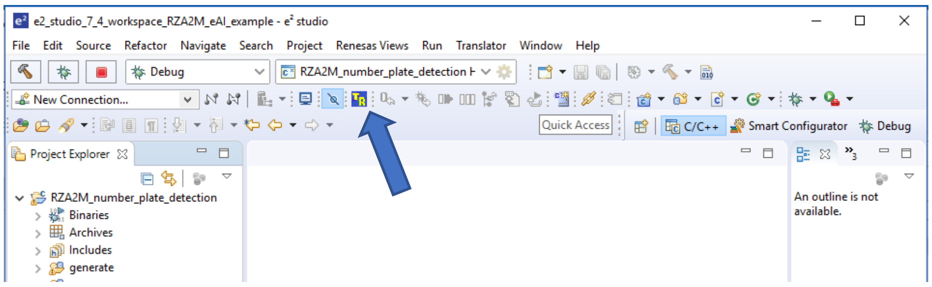

keras_to_tensorflow.py

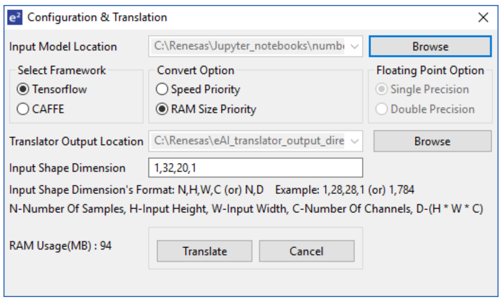
```
    sess = K.get_session()
    if FLAGS.output_meta_ckpt:
        saver = tf.train.Saver()
        saver.save(sess, str(output_fld / output_model_stem), strip_default_attrs = True)
```

Start the script with attribute --output_meta_ckpt="TRUE"

```
python keras_to_tensorflow.py \
        --input_model="<file_name>.h5" \
        --output_model="<file_name>.pb" \
        --output_meta_ckpt="TRUE"
```

The frozen model can be translated with the Renesas eAI translator (e²studio plugin).


e2_studio_7_4_workspace_RZA2M_eAI_example - e² studio
File Edit Source Refactor Navigate Search Project Renesas Views Run Translator Window Help
Debug
RZA2M_number_plate_detection
New Connection...
Quick Access
C/C++
Smart Configurator
Debug
Project Explorer
RZA2M_number_plate_detection
Binaries
Archives
Includes
generate
An outline is not available.


Configuration & Translation

Input Model Location: C:\Renesas\Jupyter_notebooks\numb — Browse

Select Framework: (●) Tensorflow ( ) CAFFE

Convert Option: ( ) Speed Priority (●) RAM Size Priority

Floating Point Option: ( ) Single Precision ( ) Double Precision

Translator Output Location: C:\Renesas\eAI_translator_output_dire — Browse

Input Shape Dimension: 1,32,20,1

Input Shape Dimension's Format: N,H,W,C (or) N,D Example: 1,28,28,1 (or) 1,784
N-Number Of Samples, H-Input Height, W-Input Width, C-Number Of Channels, D-(H * W * C)

RAM Usage(MB) : 94

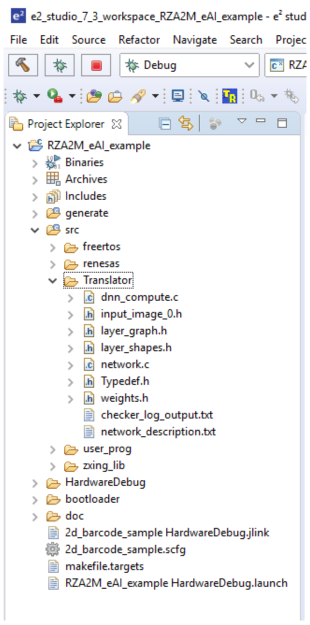
Translate Cancel

enter
“Input Model Location“,
“Translator Output Location”
and
input shape dimension

⇨ start “Translate”

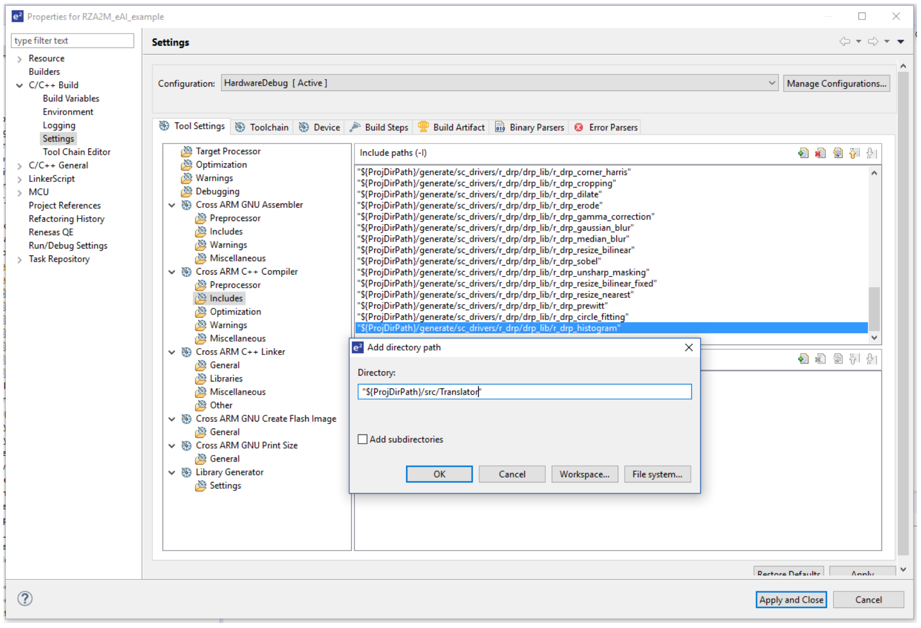
3.2. NN integration (e²studio)

Copy the eAI translator result (directory 'Translator') into the project directory



```
RZA2M_eAI_example
  Binaries
  Archives
  Includes
  generate
  src
    freertos
    renesas
    Translator
      dnn_compute.c
      input_image_0.h
      layer_graph.h
      layer_shapes.h
      network.c
      Typedef.h
      weights.h
      checker_log_output.txt
      network_description.txt
    user_prog
    zxing_lib
  HardwareDebug
  bootloader
  doc
  2d_barcode_sample HardwareDebug.jlink
  2d_barcode_sample.scfg
  makefile.targets
  RZA2M_eAI_example HardwareDebug.launch
```

Add "Translator" path to the list of Include paths
right-click at project RZA2M_eAI_example -> Properties



Properties for RZA2M_eAI_example — Settings

Configuration: HardwareDebug [Active]

Tool Settings — Cross ARM C++ Compiler — Includes

Include paths (-I)

```
"${ProjDirPath}/generate/sc_drivers/r_drp/drp_lib/r_drp_corner_harris"
"${ProjDirPath}/generate/sc_drivers/r_drp/drp_lib/r_drp_cropping"
"${ProjDirPath}/generate/sc_drivers/r_drp/drp_lib/r_drp_dilate"
"${ProjDirPath}/generate/sc_drivers/r_drp/drp_lib/r_drp_erode"
"${ProjDirPath}/generate/sc_drivers/r_drp/drp_lib/r_drp_gamma_correction"
"${ProjDirPath}/generate/sc_drivers/r_drp/drp_lib/r_drp_gaussian_blur"
"${ProjDirPath}/generate/sc_drivers/r_drp/drp_lib/r_drp_median_blur"
"${ProjDirPath}/generate/sc_drivers/r_drp/drp_lib/r_drp_resize_bilinear"
"${ProjDirPath}/generate/sc_drivers/r_drp/drp_lib/r_drp_sobel"
"${ProjDirPath}/generate/sc_drivers/r_drp/drp_lib/r_drp_unsharp_masking"
"${ProjDirPath}/generate/sc_drivers/r_drp/drp_lib/r_drp_resize_bilinear_fixed"
"${ProjDirPath}/generate/sc_drivers/r_drp/drp_lib/r_drp_resize_nearest"
"${ProjDirPath}/generate/sc_drivers/r_drp/drp_lib/r_drp_prewitt"
"${ProjDirPath}/generate/sc_drivers/r_drp/drp_lib/r_drp_circle_fitting"
"${ProjDirPath}/generate/sc_drivers/r_drp/drp_lib/r_drp_histogram"
```

Add directory path

Directory: "${ProjDirPath}/src/Translator"

Add subdirectories [ ]

"OK", "Apply and Close"

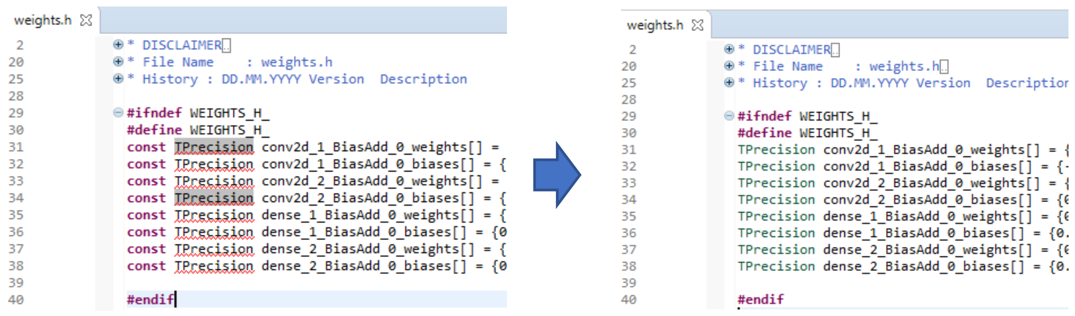
integrate neural network
e.g.:

```
#include "Typedef.h"

/* Imported global variables and functions (from other files) */
TPrecision* dnn_compute(TPrecision*);

float *prediction;
static float  work_buf_RGB_NN[32 * 32 * 3];

prediction = dnn_compute(&work_buf_RGB_NN[0]) ;
```

3.3. NN TAT optimisation (e²studio, gcc)

1. increase inference performance by loading the weights and biases into the local RAM

   Open RZA2M_eAI_example\src\Translator\weights.h
   and change the variable definition from “const TPrecision” to “TPrecision”.



weights.h

```
* DISCLAIMER
* File Name     : weights.h
* History : DD.MM.YYYY Version  Description

#ifndef WEIGHTS_H_
#define WEIGHTS_H_
const TPrecision conv2d_1_BiasAdd_0_weights[] =
const TPrecision conv2d_1_BiasAdd_0_biases[] = {
const TPrecision conv2d_2_BiasAdd_0_weights[] =
const TPrecision conv2d_2_BiasAdd_0_biases[] = {
const TPrecision dense_1_BiasAdd_0_weights[] = {
const TPrecision dense_1_BiasAdd_0_biases[] = {0
const TPrecision dense_2_BiasAdd_0_weights[] = {
const TPrecision dense_2_BiasAdd_0_biases[] = {0

#endif
```

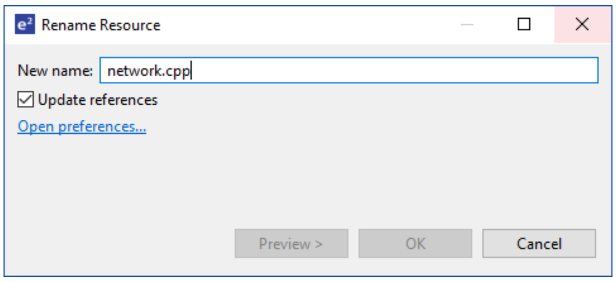
weights.h

```
* DISCLAIMER
* File Name     : weights.h
* History : DD.MM.YYYY Version  Descriptio

#ifndef WEIGHTS_H_
#define WEIGHTS_H_
TPrecision conv2d_1_BiasAdd_0_weights[] = {
TPrecision conv2d_1_BiasAdd_0_biases[] = {-
TPrecision conv2d_2_BiasAdd_0_weights[] = {
TPrecision conv2d_2_BiasAdd_0_biases[] = {
TPrecision dense_1_BiasAdd_0_weights[] = {
TPrecision dense_1_BiasAdd_0_biases[] = {0.
TPrecision dense_2_BiasAdd_0_weights[] = {
TPrecision dense_2_BiasAdd_0_biases[] = {0.

#endif
```

2. Increase inference performance by adding optimization options for loop vectorization on trees

   i. rename Translator files from .c to .cpp
   RZA2M_eAI_example\src\Translator\dnn_compute.c
   to RZA2M_eAI_example\src\Translator\dnn_compute.cpp
   and
   RZA2M_eAI_example\src\Translator\network.c
   to RZA2M_eAI_example\src\Translator\network.cpp

   right-click on the file -> Rename (tick “Update references”)


Rename Resource

New name: network.cpp

☑ Update references

ii. add Wrapper around dnn_compute.cpp

Step 1 : change function name in RZA2M_eAI_example\src\Translator\dnn_compute.cpp

add declaration:

```
TPrecision* dnn_computePre(TPrecision*);
```

modify

```
TPrecision* dnn_compute(TPrecision* conv2d_1_input_0)
```

to

```
TPrecision* dnn_computePre(TPrecision* conv2d_1_input_0)
```

Step 2 : prepare a new .cpp file RZA2M_eAI_example\src\Translator\dnnInit.cpp

content:

```
float* dnn_computePre(float* ) ;

extern "C" float* dnn_compute(float*) ;

float* dnn_compute(float* inPointer) {
        return dnn_computePre(inPointer) ;
}
```

iii. add compile options to activate loop vectorization on trees

right-click separately on file

/RZA2M_number_plate_detection/src/Translator/dnn_compute.cpp

and

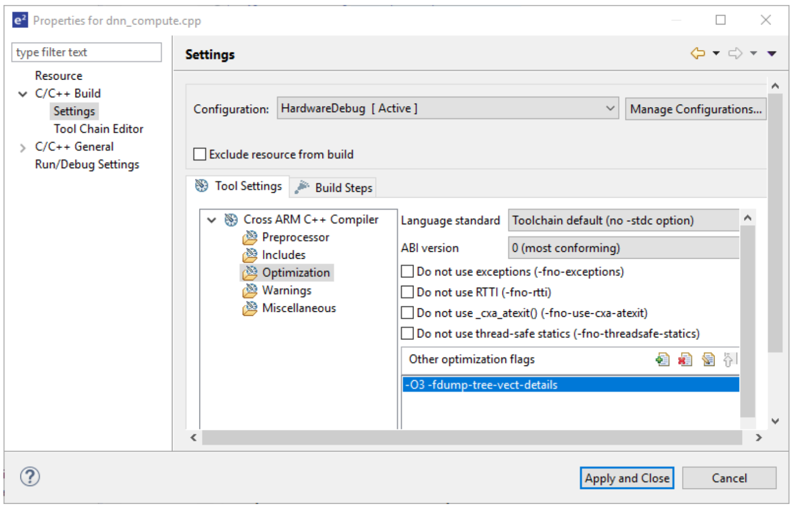
RZA2M_eAI_example\src\Translator\network.cpp

to open the “Properties” window.

Select C/C++ Build -> Settings -> Optimization

add “Other optimization flags”:

-O3 -fdump-tree-vect-details



Configuration: HardwareDebug [Active]

☐ Exclude resource from build

Language standard: Toolchain default (no -stdc option)

ABI version: 0 (most conforming)

☐ Do not use exceptions (-fno-exceptions)

☐ Do not use RTTI (-fno-rtti)

☐ Do not use _cxa_atexit() (-fno-use-cxa-atexit)

☐ Do not use thread-safe statics (-fno-threadsafe-statics)

Other optimization flags

-O3 -fdump-tree-vect-details

iv. add "__restrict" parameter to all pointer used as function parameter of file RZA2M_eAI_example\src\Translator\network.cpp
e.g.:

void convolution_without_pad(TPrecision *__restrict__ dData,const TPrecision *__restrict__ dWeights,const TPrecision *__restrict__ dBiases,TPrecision *__restrict__ dOut,TsInt *__restrict__ iShapes){

void relu(TPrecision *__restrict__ dData, TsInt iShapes)

"Clean" and "Build" the project