

Edge Computing with DSP Acceleration

Lab 3

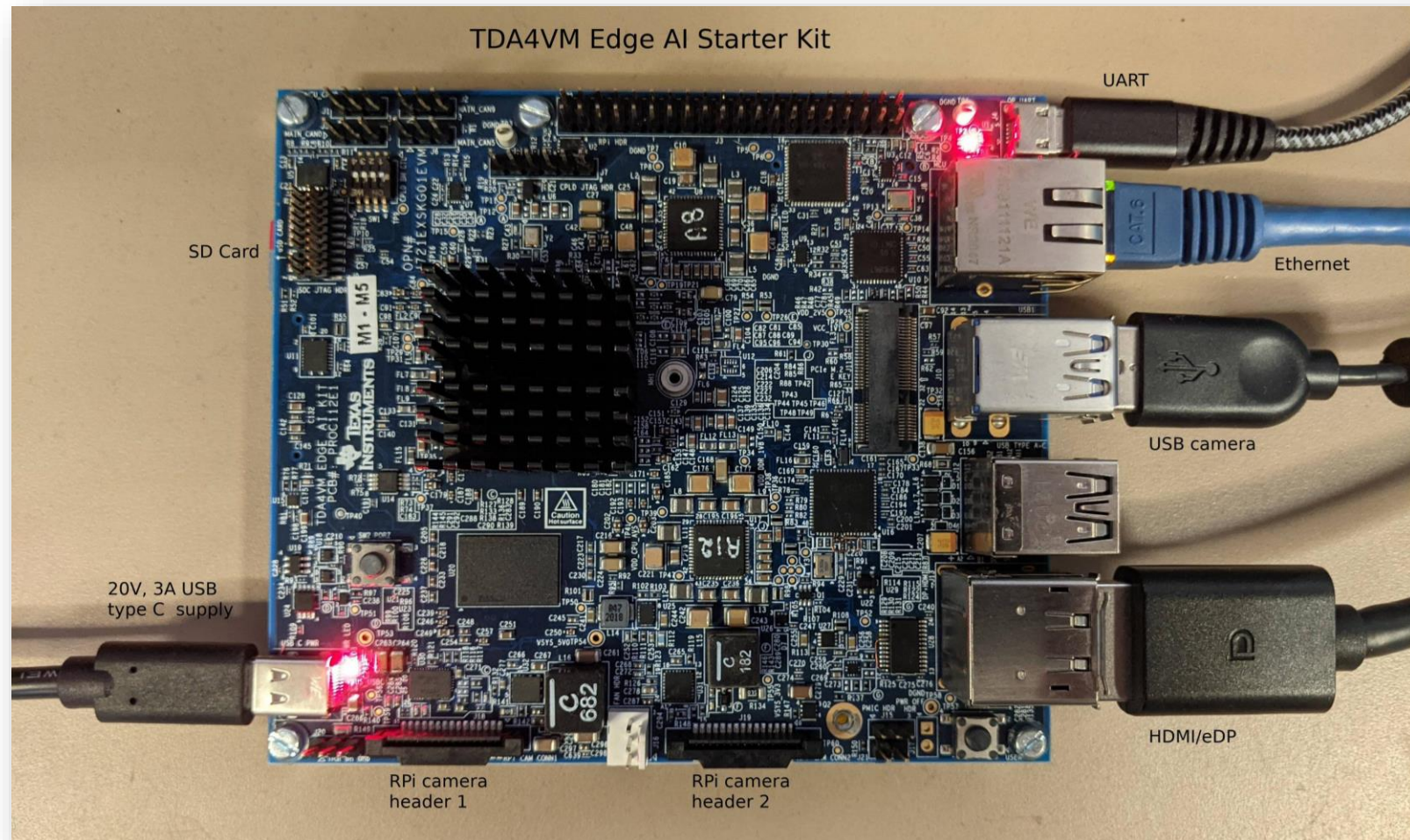
Department of Electronic and Computer Engineering
National Taiwan University of Science and Technology

學習DSP Acceleration Data Flow 環境建置

- ❖ 完成硬體加速物件偵測流程
- ❖ 實現及時輸入攝影機辨識與模型置換流程
- ❖ 紀錄DSP加速前後的FPS等各項數據



接線圖



Edge Computing Deploying Data flow

■ Accelerate Models with Open source API

- Ease of use by providing Open Source programming interface
- TIDL unsupported operators run on Cortex-A core

TensorFlow Lite models



TFLite RunTime

ONNX models

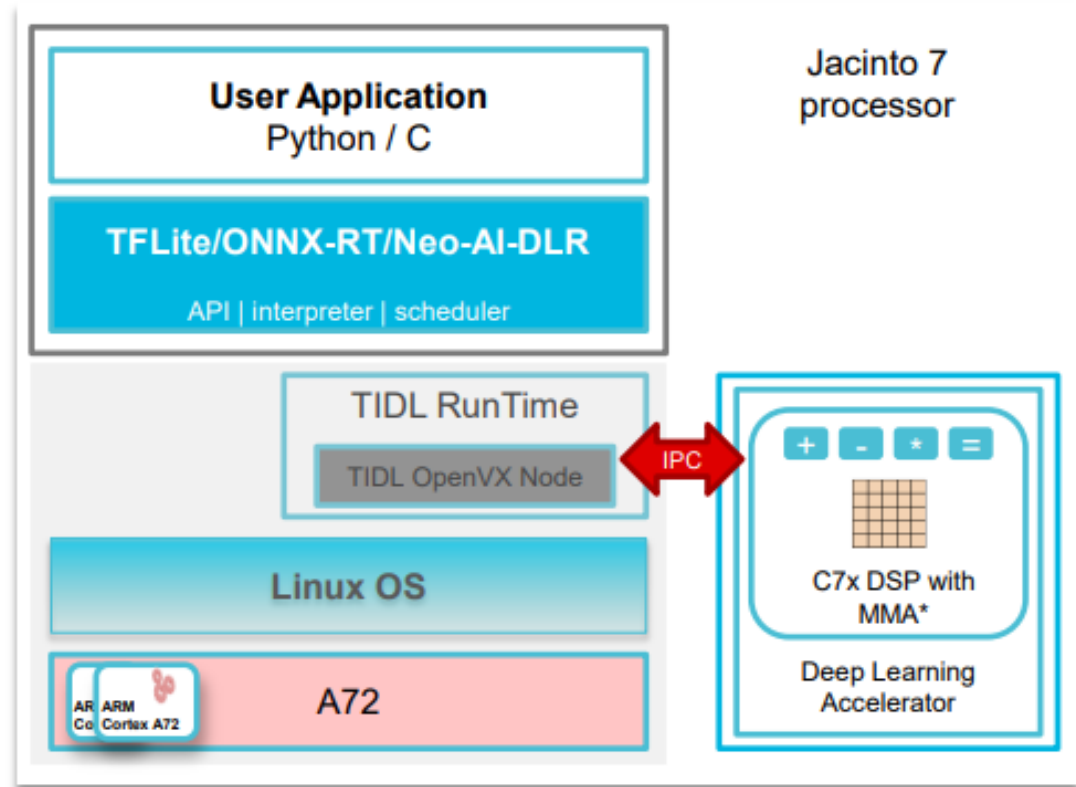


ONNX-RunTime

TVM/SageMaker
Neo compiler



Neo-AI-DLR



Object detect flow

■ TDA4VM Development Kit

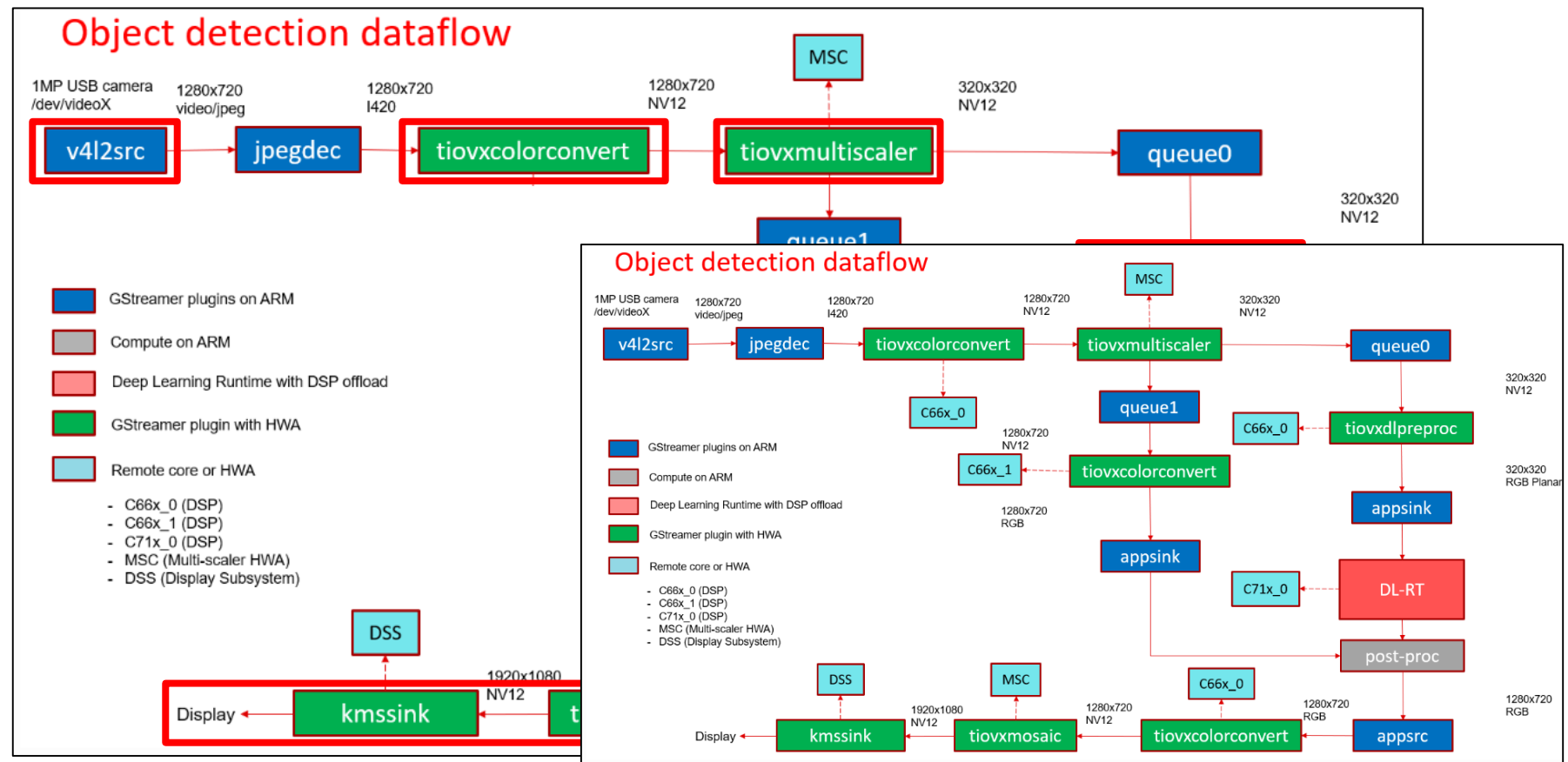
➤ Hardware acceleration data-flow pipeline for object detection demo

SDK:

<https://www.ti.com/tool/download/PROCESSOR-SDK-LINUX-SK-TDA4VM>

Fetching software:

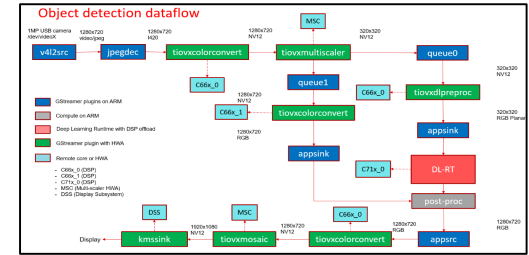
<https://www.balena.io/etcher>



物件偵測流程

➤ 匯入各Block並建立register連接

➤ 建立Gstreamer之整體流程



<https://github.com/TexasInstruments/edgeai-gst-plugins/blob/main/ext/tiovx/gsttiovx.c>

```

1  #ifdef HAVE_CONFIG_H
2  #include <config.h>
3  #endif
4
5  #include <gst/gst.h>
6
7  #include "gsttiovxdelay.h"
8  #include "gsttiovxdemux.h"
9  #include "gsttiovxisp.h"
10 #include "gsttiovxldc.h"
11 #include "gsttiovxmemalloc.h"
12 #include "gsttiovxmosaic.h"
13 #include "gsttiovxmultiscaler.h"
14 #include "gsttiovmux.h"
15 #include "gsttiovxpyramid.h"
16 #include "gsttiovxdlcolorconvert.h"
17
18 #if defined(DL_PLUGINS)
19 #include "gsttiovxdlpreproc.h"
20 #endif
21
22 #if defined(SOC_J721E) || defined(SOC_J721S2) || defined(SOC_J784S4) || defined(SOC_J722S)
23 #include "gsttiovxcolorconvert.h"
24 #include "gsttiovxdoof.h"
25 #include "gsttiovxdoofviz.h"
26 #include "gsttiovxdsde.h"
27 #include "gsttiovxdsdeviz.h"
28 #endif
29
30 #include "gst-lib/gst/tiovx/gsttiovxutils.h"

```

```

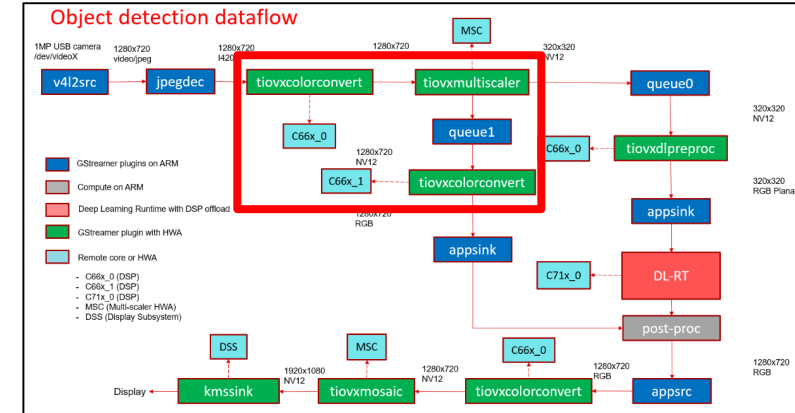
53 if defined(SOC_J721E) || defined(SOC_J721S2) || defined(SOC_J784S4) || defined(SOC_J722S)
54 ret = gst_element_register (plugin, "tiovxcolorconvert", GST_RANK_NONE,
55 | GST_TYPE_TIOVX_COLOR_CONVERT);
56 if (!ret) {
57     GST_ERROR ("Failed to register the tiovxcolorconvert element");
58     goto out;
59 }
60
61 ret = gst_element_register (plugin, "tiovxdoof", GST_RANK_NONE,
62 | GST_TYPE_TIOVX_DOF);
63 if (!ret) {
64     GST_ERROR ("Failed to register the tiovxdoof element");
65     goto out;
66 }
67
68 ret = gst_element_register (plugin, "tiovxdoofviz", GST_RANK_NONE,
69 | GST_TYPE_TIOVX_DOF_VIZ);
70 if (!ret) {
71     GST_ERROR ("Failed to register the tiovxdoofviz element");
72     goto out;
73 }
74
75 ret = gst_element_register (plugin, "tiovxdsde", GST_RANK_NONE,
76 | GST_TYPE_TIOVX_DSDE);
77 if (!ret) {
78     GST_ERROR ("Failed to register the tiovxdsde element");
79     goto out;
80 }
81
82 ret = gst_element_register (plugin, "tiovxdsdeviz", GST_RANK_NONE,
83 | GST_TYPE_TIOVX_DSDE_VIZ);
84 if (!ret) {
85     GST_ERROR ("Failed to register the tiovxdsdeviz element");
86     goto out;
87 }
88 #endif
89
90 gst_tiovx_init_debug ();

```

DSP Source Code 部分掛載

➤ Colorconvert掛載DSP流程

➤ C66x DSP Source code



1

- 定義色彩轉換元件相關資訊

2

- 初始化 GStreamer 元件

3

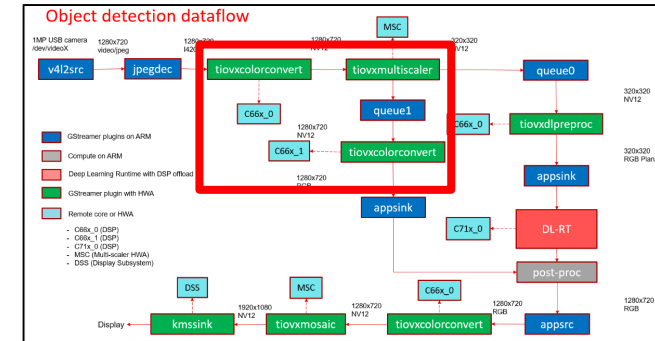
- 創建轉換輸入輸出 caps 格式與轉換圖像

4

- 掛載C66x Target 釋放整體buffer

TIOVX Gstreamer Component

- 定義轉換格式與圖像
- ColorConvert Block DSP Code



```

1  /* Formats definition */
2  #define TIOVX_COLOR_CONVERT_SUPPORTED_FORMATS_SRC "{RGB, RGBx, NV12, I420, Y444}"
3  #define TIOVX_COLOR_CONVERT_SUPPORTED_FORMATS_SINK "{RGB, RGBx, NV12, NV21, UYVY, YUY2, I420}"
4  #define TIOVX_COLOR_CONVERT_SUPPORTED_WIDTH "[1 , 8192]"
5  #define TIOVX_COLOR_CONVERT_SUPPORTED_HEIGHT "[1 , 8192]"
6  #define TIOVX_COLOR_CONVERT_SUPPORTED_CHANNELS "[1 , 16]"
7
8  /* Src caps */
9  #define TIOVX_COLOR_CONVERT_STATIC_CAPS_SRC \
10 "video/x-raw, " \
11 "format = (string) " TIOVX_COLOR_CONVERT_SUPPORTED_FORMATS_SRC ", " \
12 "width = " TIOVX_COLOR_CONVERT_SUPPORTED_WIDTH ", " \
13 "height = " TIOVX_COLOR_CONVERT_SUPPORTED_HEIGHT \
14 "; " \
15 "video/x-raw(" GST_CAPS_FEATURE_BATCHED_MEMORY ")," \
16 "format = (string) " TIOVX_COLOR_CONVERT_SUPPORTED_FORMATS_SRC ", " \
17 "width = " TIOVX_COLOR_CONVERT_SUPPORTED_WIDTH ", " \
18 "height = " TIOVX_COLOR_CONVERT_SUPPORTED_HEIGHT ", " \
19 "num-channels = " TIOVX_COLOR_CONVERT_SUPPORTED_CHANNELS \
20

```

```

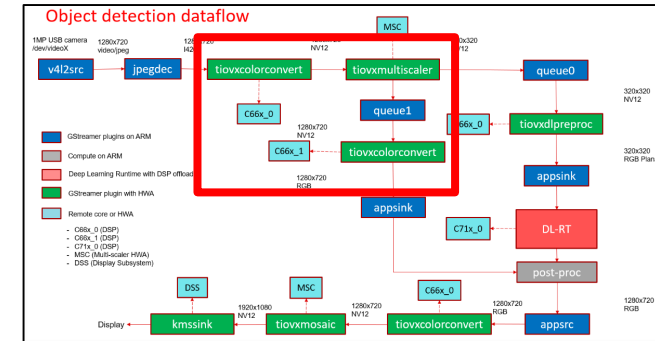
21 /* Sink caps */
22 #define TIOVX_COLOR_CONVERT_STATIC_CAPS_SINK \
23 "video/x-raw, " \
24 "format = (string) " TIOVX_COLOR_CONVERT_SUPPORTED_FORMATS_SINK ", " \
25 "width = " TIOVX_COLOR_CONVERT_SUPPORTED_WIDTH ", " \
26 "height = " TIOVX_COLOR_CONVERT_SUPPORTED_HEIGHT \
27 "; " \
28 "video/x-raw(" GST_CAPS_FEATURE_BATCHED_MEMORY ")," \
29 "format = (string) " TIOVX_COLOR_CONVERT_SUPPORTED_FORMATS_SINK ", " \
30 "width = " TIOVX_COLOR_CONVERT_SUPPORTED_WIDTH ", " \
31 "height = " TIOVX_COLOR_CONVERT_SUPPORTED_HEIGHT ", " \
32 "num-channels = " TIOVX_COLOR_CONVERT_SUPPORTED_CHANNELS \
33
34 /* Pads definitions */
35 static GstStaticPadTemplate sink_template = GST_STATIC_PAD_TEMPLATE ("sink",
36 GST_PAD_SINK,
37 GST_PAD_ALWAYS,
38 GST_STATIC_CAPS (TIOVX_COLOR_CONVERT_STATIC_CAPS_SINK)
39 );
40
41 static GstStaticPadTemplate src_template = GST_STATIC_PAD_TEMPLATE ("src",
42 GST_PAD_SRC,
43 GST_PAD_ALWAYS,
44 GST_STATIC_CAPS (TIOVX_COLOR_CONVERT_STATIC_CAPS_SRC)
45 );

```

定義source影像格式等等
I420 -> NV12

TIOVX Gstreamer Component

- 初始化所有class與target元件
- ColorConvert Block DSP Code



```

1  /* Initialize the plugin's class */
2  static void
3  gst_tiovx_color_convert_class_init (GstTIOVXColorconvertClass * klass)
4  {
5      GObjectClass *gobject_class = NULL;
6      GstBaseTransformClass *gstbasetransform_class = NULL;
7      GstElementClass *gstelement_class = NULL;
8      GstTIOVXSisoClass *gsttiovxsiso_class = NULL;
9
10     gobject_class = G_OBJECT_CLASS (klass);
11     gstbasetransform_class = GST_BASE_TRANSFORM_CLASS (klass);
12     gstelement_class = GST_ELEMENT_CLASS (klass);
13     gsttiovxsiso_class = GST_TIOVX_SISO_CLASS (klass);
14
15     gst_element_class_set_details_simple (gstelement_class,
16         "TIOVX ColorConvert",
17         "Filter/Converter/Video",
18         "Converts video from one colorspace to another using the TIOVX Modules API",
19         "RidgeRun support@ridgerun.com");
20
21     gst_element_class_add_pad_template (gstelement_class,
22         gst_static_pad_template_get (&src_template));
23     gst_element_class_add_pad_template (gstelement_class,
24         gst_static_pad_template_get (&sink_template));
25

```

```

26     gobject_class->set_property = gst_tiovx_color_convert_set_property;
27     gobject_class->get_property = gst_tiovx_color_convert_get_property;
28
29     g_object_class_install_property (gobject_class, PROP_TARGET,
30         g_param_spec_enum ("target", "Target",
31             "TIOVX target to use by this element",
32             GST_TYPE_TIOVX_COLOR_CONVERT_TARGET,
33             DEFAULT_TIOVX_COLOR_CONVERT_TARGET,
34             G_PARAM_READWRITE | G_PARAM_CONTROLLABLE | G_PARAM_STATIC_STRINGS));
35
36     gstbasetransform_class->transform_caps =
37         GST_DEBUG_FUNCPTR (gst_tiovx_color_convert_transform_caps);
38     /* Disable processing if input & output caps are equal, i.e., no format conversion */
39     gstbasetransform_class->passthrough_on_same_caps = TRUE;
40     gstbasetransform_class->transform_ip_on_passthrough = FALSE;
41
42     gsttiovxsiso_class->init_module =
43         GST_DEBUG_FUNCPTR (gst_tiovx_color_convert_init_module);
44     gsttiovxsiso_class->create_graph =
45         GST_DEBUG_FUNCPTR (gst_tiovx_color_convert_create_graph);
46     gsttiovxsiso_class->get_node_info =
47         GST_DEBUG_FUNCPTR (gst_tiovx_color_convert_get_node_info);
48     gsttiovxsiso_class->release_buffer =
49         GST_DEBUG_FUNCPTR (gst_tiovx_color_convert_release_buffer);
50     gsttiovxsiso_class->deinit_module =
51         GST_DEBUG_FUNCPTR (gst_tiovx_color_convert_deinit_module);
52     gsttiovxsiso_class->compare_caps =
53         GST_DEBUG_FUNCPTR (gst_tiovx_color_convert_compare_caps);
54
55     GST_DEBUG_CATEGORY_INIT (gst_tiovx_color_convert_debug,
56         "tiouvcolorconvert", 0, "TIOVX ColorConvert element");
57

```

Initialize各類別物件

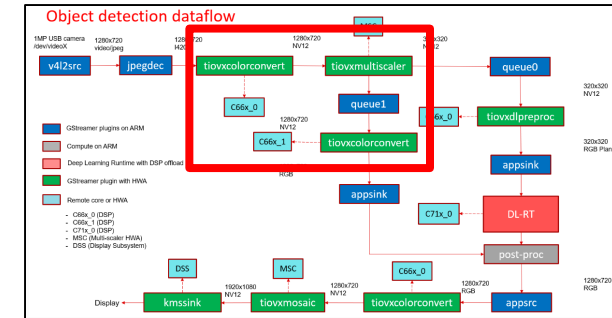
TIOVX Gstreamer Component

- 找到輸入影像的format
- ColorConvert Block DSP Code

```

1 static gboolean
2 get_formats (const GValue * input_formats, GValue * output_formats,
3             AppendFormatFunc cb)
4 {
5     gint i = 0;
6     gboolean ret = TRUE;
7     gint size = 0;
8
9     g_return_val_if_fail (input_formats, FALSE);
10    g_return_val_if_fail (output_formats, FALSE);
11    g_return_val_if_fail (GST_VALUE_HOLDS_LIST (output_formats), FALSE);
12
13    size =
14        GST_VALUE_HOLDS_LIST (input_formats) ?
15        gst_value_list_get_size (input_formats) : 1;
16
17    for (i = 0; i < size; i++) {
18        const GValue *value = NULL;
19        const gchar *format_name = NULL;
20        GstVideoFormat format = GST_VIDEO_FORMAT_UNKNOWN;
21
22        if (GST_VALUE_HOLDS_LIST (input_formats)) {
23            value = gst_value_list_get_value (input_formats, i);
24        } else {
25            value = input_formats;
26        }
27
28        format_name = g_value_get_string (value);
29        format = gst_video_format_from_string (format_name);
30
31        if (FALSE == cb (format, output_formats)) {
32            ret = FALSE;
33            break;
34        }
35    }
36
37    return ret;
38 }

```



以For迴圈獲取
各輸入之格式

TIOVX Gstreamer Component

➤ 影像輸出入格式轉換

➤ ColorConvert Block DSP Code

```

1 static GstCaps *
2 gst_tiovx_color_convert_transform_caps (GstBaseTransform * base,
3     GstPadDirection direction, GstCaps * caps, GstCaps * filter)
4 {
5     GstTIOVXColorconvert *self = GST_TIOVX_COLOR_CONVERT (base);
6     GstCaps *result_caps = NULL;
7     gint i = 0;
8     AppendFormatFunc func;
9
10    GST_DEBUG_OBJECT (self, "Transforming caps on %s:\ncaps: %"
11        GST_PTR_FORMAT "\nfilter: %" GST_PTR_FORMAT,
12        GST_PAD_SRC == direction ? "src" : "sink", caps, filter);
13
14    result_caps = gst_caps_copy (caps);
15
16    for (i = 0; i < gst_caps_get_size (result_caps); i++) {
17        GstStructure *st = gst_caps_get_structure (result_caps, i);
18        GstStructure *st_copy = gst_structure_copy (st);
19        const GValue *input_formats = gst_structure_get_value (st, "format");
20        GValue output_formats = G_VALUE_INIT;
21        const GValue *tmp_val = NULL;
22
23        g_value_init (&output_formats, GST_TYPE_LIST);
24
25        if (GST_PAD_SINK == direction) {
26            func = append_src_formats;
27        } else {
28            func = append_sink_formats;
29        }
30
31        get_formats (input_formats, &output_formats, func);
32        gst_structure_remove_all_fields (st);
33    }

```

```

get_formats (input_formats, &output_formats, func);
gst_structure_remove_all_fields (st);

/* Set tranformed format */
gst_structure_set_value (st, "format", &output_formats);
g_value_unset (&output_formats);

/* Copy other fields as it is */
tmp_val = gst_structure_get_value (st_copy, "width");
if (tmp_val != NULL) {
    gst_structure_set_value (st, "width", tmp_val);
}

tmp_val = gst_structure_get_value (st_copy, "height");
if (tmp_val != NULL) {
    gst_structure_set_value (st, "height", tmp_val);
}

tmp_val = gst_structure_get_value (st_copy, "framerate");
if (tmp_val != NULL) {
    gst_structure_set_value (st, "framerate", tmp_val);
}

gst_structure_free (st_copy);
}

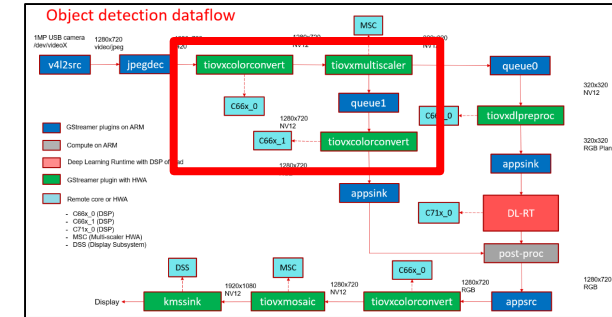
if (filter) {
    GstCaps *tmp = result_caps;
    result_caps = gst_caps_intersect (result_caps, filter);
    gst_caps_unref (tmp);
}

GST_DEBUG_OBJECT (self, "Resulting caps are %" GST_PTR_FORMAT, result_caps);

return result_caps;
}

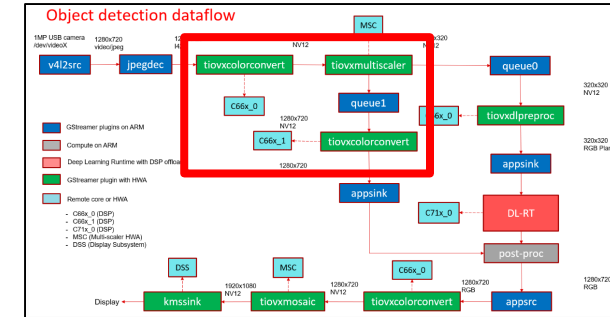
```

以For迴圈轉換
各輸入之格式
並定義長寬高



TIOVX Gstreamer Component

- 影像圖像輸出入轉換
- ColorConvert Block DSP Code



```

1 static gboolean
2 gst_tiovx_color_convert_create_graph (GstTIOVXSiso * trans, vx_context context,
3   vx_graph graph)
4 {
5     GstTIOVXColorconvert *self = NULL;
6     vx_status status = VX_SUCCESS;
7     const char *target = NULL;
8     gboolean ret = FALSE;
9
10    g_return_val_if_fail (trans, FALSE);
11    g_return_val_if_fail (VX_SUCCESS == vxGetStatus ((vx_reference) context),
12      FALSE);
13    g_return_val_if_fail (VX_SUCCESS == vxGetStatus ((vx_reference) graph),
14      FALSE);
15
16    self = GST_TIOVX_COLOR_CONVERT (trans);
17
18    GST_INFO_OBJECT (self, "Create graph");
19
20    GST_OBJECT_LOCK (GST_OBJECT (self));
21    target = target_id_to_target_name (self->target_id);
22    GST_OBJECT_UNLOCK (GST_OBJECT (self));
  
```

創建圖像
轉換物件

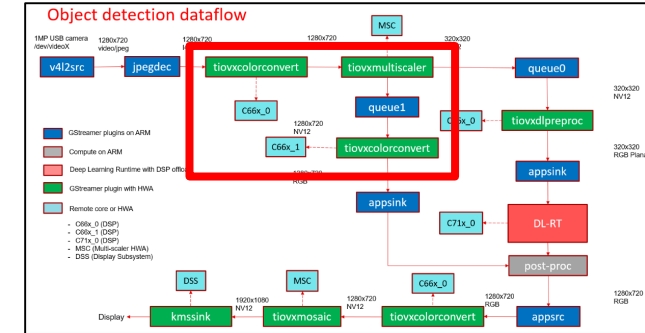
```

23 if (NULL == target) {
24     GST_ERROR_OBJECT (self, "TIOVX target selection failed");
25     goto out;
26 }
27
28 GST_INFO_OBJECT (self, "TIOVX Target to use: %s", target);
29
30 status = tiouv_color_convert_module_create (graph, &self->obj, NULL, target);
31 if (VX_SUCCESS != status) {
32     GST_ERROR_OBJECT (self, "Create graph failed with error: %d", status);
33     goto out;
34 }
35
36 ret = TRUE;
37
38 out:
39 return ret;
40 }
  
```

Target ID指定為C66x元件
通過編譯時TIDL工具調用

DSP Source Code 部分掛載

- 定義編譯硬體格式與Target DSP
- C66x DSP Source code



```
static GType
gst_tiovx_color_convert_target_get_type (void)
{
    static GType target_type = 0;

    static const GEnumValue targets[] = {
#ifdef SOC_J721E
        {TIVX_CPU_ID_DSP1, "DSP instance 1, assigned to C66_0 core",
         TIVX_TARGET_DSP1},
        {TIVX_CPU_ID_DSP2, "DSP instance 2, assigned to C66_1 core",
         TIVX_TARGET_DSP2},
#elif defined(SOC_J721S2) || defined(SOC_J784S4) || defined(SOC_J722S)
        {TIVX_CPU_ID_DSP1, "DSP instance 1, assigned to C7_2 core",
         TIVX_TARGET_DSP1},
#endif
        {0, NULL, NULL},
    };
};
```

建立物件目標

Target ID 為 DSP C66

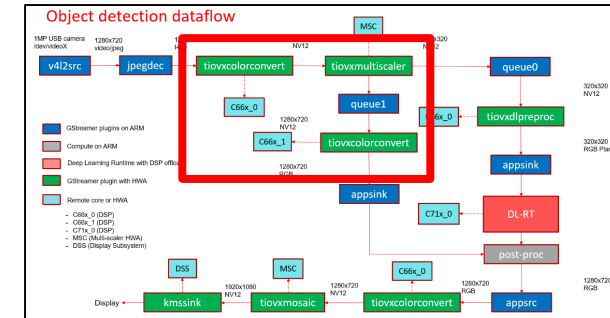
C66x DSP掛載成功

編譯TIDL調用

```
1 static const gchar *
2 target_id_to_target_name (gint target_id)
3 {
4     GType type = G_TYPE_NONE;
5     GEnumClass *enum_class = NULL;
6     GEnumValue *enum_value = NULL;
7     const gchar *value_nick = NULL;
8
9     type = gst_tiovx_color_convert_target_get_type ();
10    enum_class = G_ENUM_CLASS (g_type_class_ref (type));
11    enum_value = g_enum_get_value (enum_class, target_id);
12    value_nick = enum_value->value_nick;
13    g_type_class_unref (enum_class);
14
15    return value_nick;
16 }
```


C66x DSP Source Code 掛載

- Buffer釋放與Target宣告
- ColorConvert Block DSP Code



```

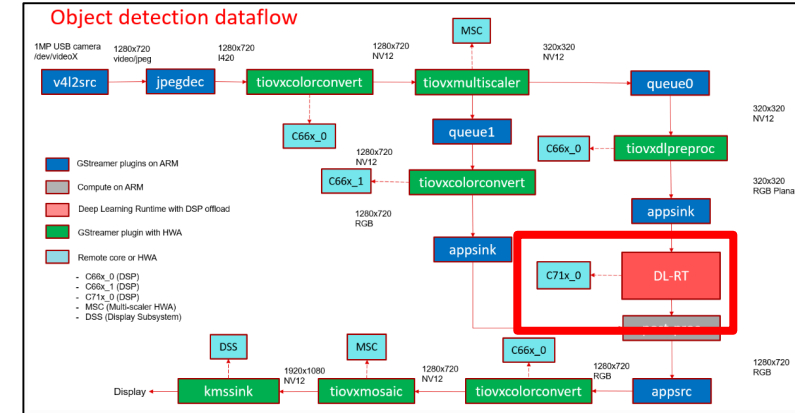
1  static gboolean
2  gst_tiovx_color_convert_release_buffer (GstTIOVXSiso * trans)
3  {
4      GstTIOVXColorconvert *self = NULL;
5      vx_status status = VX_SUCCESS;
6
7      g_return_val_if_fail (trans, FALSE);
8
9      self = GST_TIOVX_COLOR_CONVERT (trans);
10
11     GST_INFO_OBJECT (self, "Release buffer");
12
13     status = tiovx_color_convert_module_release_buffers (&self->obj);
14     if (VX_SUCCESS != status) {
15         GST_ERROR_OBJECT (self, "Release buffer failed with error: %d", status);
16         return FALSE;
17     }
18
19     return TRUE;
20 }

```

釋放各buffer緩衝

DSP Source Code 部分掛載

- DL-RT之神經網路加速
- C71x DSP Source code



1

- 初始化
GStreamer 元
件

2

- 讀取模型並
建立整體
DSP連接

3

- 建立輸入輸出
整體參數結構
調整

4

- 定義target 的
DSP目標在編譯
時實現

TIOVX Gstreamer Component

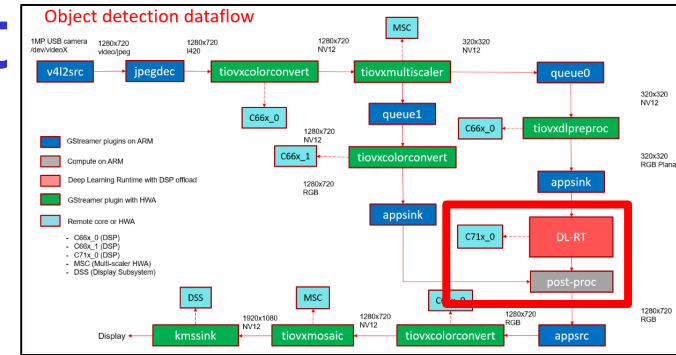
➤ 初始化 GStreamer 元件

➤ DL-RT Source code

```

1  /* Initialize the plugin's class */
2  static void
3  gst_ti_dl_inferer_class_init (GstTIDLInfererClass * klass)
4  {
5      GObjectClass *
6      | gobject_class = NULL;
7      GstBaseTransformClass *
8      | gstbasetransform_class = NULL;
9      GstElementClass *
10     | gstelement_class = NULL;
11
12     gobject_class = (GObjectClass *) klass;
13     gstbasetransform_class = (GstBaseTransformClass *) klass;
14     gstelement_class = (GstElementClass *) klass;
15
16     gst_element_class_set_details_simple (gstelement_class,
17     | "TI DL Inferer",
18     | "Filter/Converter/Tensor",
19     | "Compute the DL inference of input tensor for model specified",
20     | "Rahul T R <r-ravikumar@ti.com>");
21
22     gst_element_class_add_pad_template (gstelement_class,
23     | gst_static_pad_template_get (&src_template));
24     gst_element_class_add_pad_template (gstelement_class,
25     | gst_static_pad_template_get (&sink_template));
26
27     gobject_class->set_property = gst_ti_dl_inferer_set_property;
28     gobject_class->get_property = gst_ti_dl_inferer_get_property;

```



```

1  #ifdef ENABLE_TIDL
2      g_object_class_install_property (gobject_class, PROP_TARGET,
3      | g_param_spec_enum ("target", "Target",
4      | "C7x target to offload the inference",
5      | GST_TYPE_TI_DL_INFERER_TARGET,
6      | DEFAULT_TI_DL_INFERER_TARGET,
7      | (GParamFlags)(G_PARAM_READWRITE | GST_PARAM_CONTROLLABLE
8      | | G_PARAM_STATIC_STRINGS));
9  #endif //ENABLE_TIDL

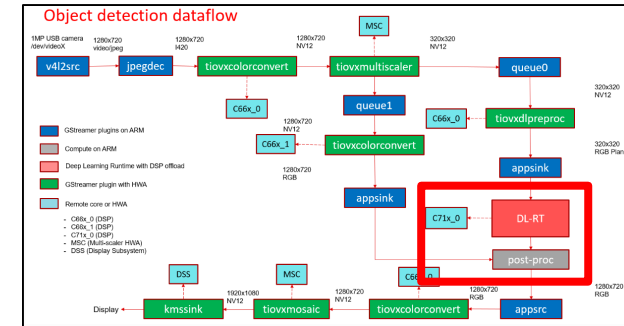
```

Initialize整體資料

<https://github.com/TexasInstruments/edgeai-gst-plugins/blob/main/ext/ti/gsttidlinferer.cpp#L270>

TIOVX Gstreamer Component

- 匯入gstreamer等辨識套件
- DL-RT神經網路Block DSP Code



```

1  import config_parser
2  import gst_wrapper
3  from gst_element_map import gst_element_map
4  from edgeai_dl_inferer import ModelConfig
5  from infer_pipe import InferPipe
6  import utils
7  import sys
8  import os
9  import time
10
11 class EdgeAIDemo:
12     """
13     Abstract the functionality required for the Edge AI demo.
14     Creates Input, Model, Output and Flow objects. Sets up infer pipes
15     for each flow and starts the infer pipes
16     """
17     C7_CORE_ID_INDEX = 0
18
19     def __init__(self, config):
20         """
21         Constructor of EdgeAIDemo class
22         Args:
23             config: Dictionary of params passed from config file
24         """
25         self.config = config
26         self.models = {}
27         self.inputs = {}
28         self.outputs = {}
29         self.flows = []
30         self.infer_pipes = []
31         self.title = config["title"]

```

TIOVX Gstreamer Component

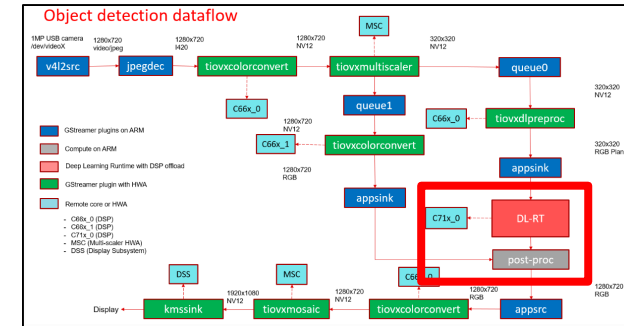
➤ 模型建立與DSP選用

➤ DL-RT神經網路Block DSP Code

```

1 # Parse Input/Model/Output Objects
2 if model not in self.models:
3     model_config = config["models"][model]
4     model_path = model_config["model_path"]
5     # Make model Config. This class is present in edgeai dl inferer
6     enable_tidl = False
7     core_id = 1
8     if (gst_element_map['inferer']['target'] == 'dsp'):
9         enable_tidl = True
10        if 'core-id' in gst_element_map['inferer']:
11            core_id = gst_element_map['inferer']['core-id'][EdgeAIDemo.C7_CORE_ID_INDEX]
12            EdgeAIDemo.C7_CORE_ID_INDEX += 1
13            if EdgeAIDemo.C7_CORE_ID_INDEX >= len(gst_element_map['inferer']['core-id']):
14                EdgeAIDemo.C7_CORE_ID_INDEX = 0
15        elif (gst_element_map['inferer']['target'] != 'arm'):
16            print("[WARNING] Invalid target specified for inferer. Defaulting to ARM.")
17
18        model_obj = ModelConfig(model_path,enable_tidl,core_id)
19
20        # Initialize the runtime
21        model_obj.create_runtime()
22
23        # task specific params
24        if "alpha" in model_config:
25            model_obj.alpha = model_config["alpha"]
26        if "viz_threshold" in model_config:
27            model_obj.viz_threshold = model_config["viz_threshold"]
28        if "topN" in model_config:
29            model_obj.topN = model_config["topN"]
30
31        self.models[model] = model_obj

```



選取target是否要掛載DSP
如不要則使用ARM Cortex
並ENABLE TIDL

創建模型與參數設定

https://github.com/TexasInstruments/edgeai-gst-apps/blob/main/apps_python/edge_ai_class.py#L89

DSP Source Code 部分掛載

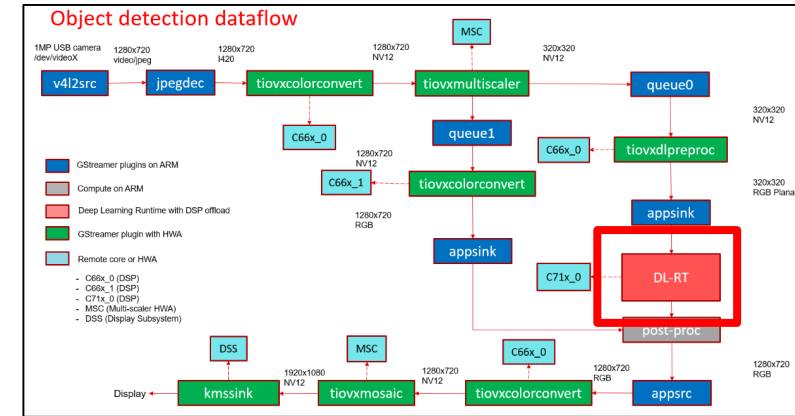
➤ 定義輸入出資料數據結構

➤ C71x DSP Source code 3

```

1  #ifndef ENABLE_TIDL
2      self->context = vxCreateContext ();
3
4      dim_sizes[0] = self->input_buffs[0]->shape[1];
5      dim_sizes[1] = self->input_buffs[0]->shape[2];
6      dim_sizes[2] = self->input_buffs[0]->shape[3];
7      self->input_ref = vxCreateTensor (self->context, NUM_TENSOR_DIMS, dim_sizes,
8          self->input_buffs[0]->type, 0);
9
10 #else
11     self->input_size = getTypeSize (self->input_buffs[0]->type);
12     for (gint i=0; i < self->input_buffs[0]->dim; i++) {
13         self->input_size *= self->input_buffs[0]->shape[i];
14     }
15 #endif //ENABLE_TIDL
16
17     status = gst_ti_dl_inferer_set_output_params(self);
18     if (status < 0) {
19         GST_ERROR_OBJECT (self, "Failed to set output params");
20         goto exit;
21     }
22
23 #ifndef ENABLE_TIDL
24     dim_sizes[0] = self->output_width;
25     dim_sizes[1] = self->output_height;
26     dim_sizes[2] = 1;
27
28     self->output_ref = vxCreateTensor (self->context, NUM_TENSOR_DIMS,
29         dim_sizes, DLIInferType UInt8, 0);
30 #else
31     self->output_width = ARM_MAX_OUTPUT_WIDTH;
32     self->output_height = ARM_MAX_OUTPUT_HEIGHT;
33     self->output_size = self->output_width * self->output_height;
34 #endif //ENABLE_TIDL
35
36     GST_LOG_OBJECT (self, "output width = %u", self->output_width);
37     GST_LOG_OBJECT (self, "output height = %u", self->output_height);
38 }
39
40 result_caps = gst_caps_from_string (TI_DL_INFERER_STATIC_CAPS);

```



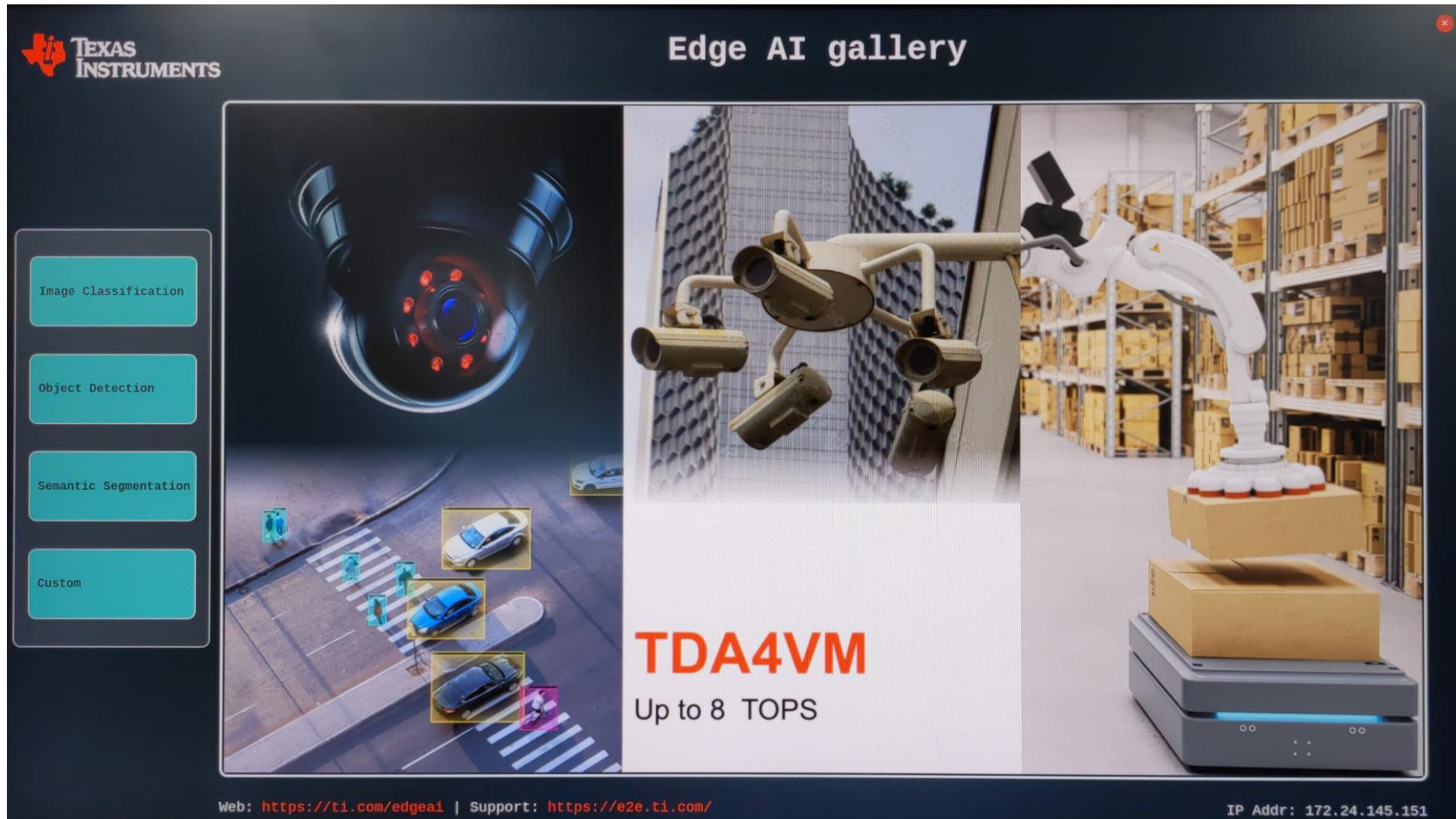
建立輸入輸出shape與
DSP掛載格式和OPENVX

如不啟用則以ARM實現

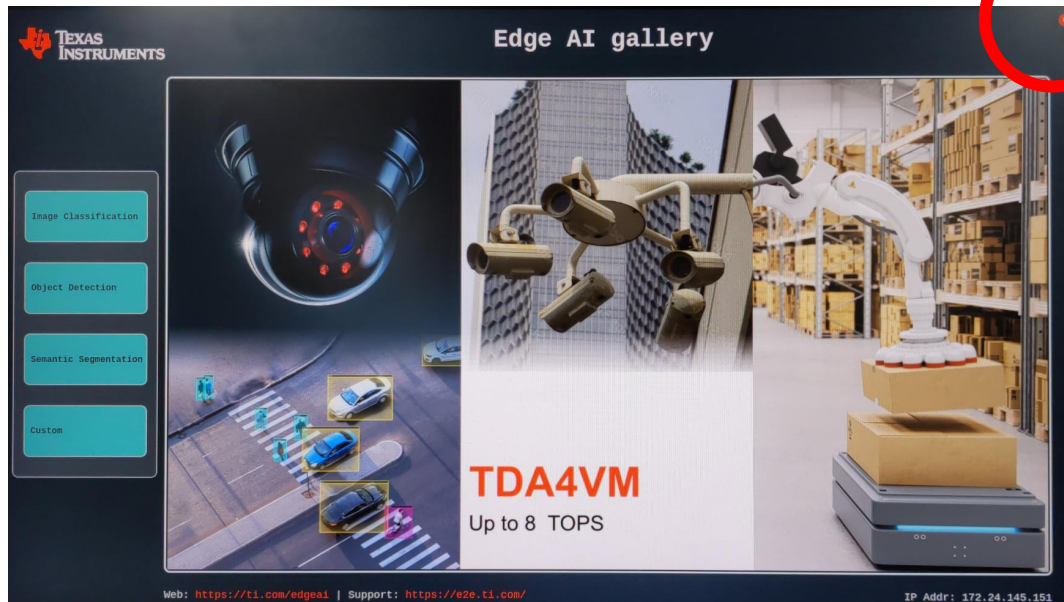
<https://github.com/TexasInstruments/edgeai-gst-plugins/blob/main/ext/ti/gsttidlinferer.cpp#L576>

進入終端機

開機畫面



終端機



1. 按右上角x鍵離開圖形選單
2. 按Ctrl+Alt+F1
進入終端機

終端機

```
Arago Project Arago Project  
Arago Project tda4um-sk -  
Arago 2023.04 tda4um-sk -  
tda4um-sk login: root  
USB Camera 0 detected  
  device = /dev/video-usb-can0  
  format = jpeg  
root@tda4um-sk:/opt/edgeai-gst-apps#
```

進入終端機後，
輸入 **root** 登錄，
並確認讀取到USB相機

Tips: 按Tab會自動補全指令或
跑出可能指令

確認相機設備節點與規格

確認相機設備節點與規格

確認相機設備節點

`v4l2-ctl --list-devices`

此指令可列出系統上所有可用的 V4L2 視訊設備

目前連接的相機為 C270 HD WEBCAM (全彩可見光相機)

可得知有三個設備節點，每個節點皆對應不同功能

`/dev/video2`

`/dev/video3`

`/dev/media2`

```
root@tda4vm-sk:/opt/edgeai-gst-apps/configs v4l2-ctl --list-devices
[ 455.337274] usb 1-1.1: reset high-speed USB device number 3 using xhci-hcd
TI-CSI2RX (platform:4500000.ticsi2rx):
/dev/media0

TI-CSI2RX (platform:4510000.ticsi2rx):
/dev/media1

vxd-dec (platform:vxd-dec):
/dev/video0

vxe-enc (platform:vxe-enc):
/dev/video1

C270 HD WEBCAM (usb-xhci-hcd.6.auto-1.1):
/dev/video2
/dev/video3
/dev/media2
```

確認相機設備節點與規格

查看設備節點功能

/dev/video2: 功能為影像/視訊擷取

/dev/video3: 資料Metadata

/dev/media2: 調整Video Pipeline用

```
v4l2-ctl -d2 --list-formats-ext
```

此指令可列出所有支援的影像格式及尺寸、FPS

(-d2 = /dev/video2)

```
ioctl: VIDIOC_ENUM_FMT
Type: Video Capture

[0]: 'NV12' (Y/UV 4:2:0)
    Size: Discrete 640x480
        Interval: Discrete 0.033s (30.000 fps)
        Interval: Discrete 0.036s (28.000 fps)
        Interval: Discrete 0.038s (26.000 fps)
        Interval: Discrete 0.042s (24.000 fps)
        Interval: Discrete 0.043s (23.000 fps)
        Interval: Discrete 0.048s (21.000 fps)
        Interval: Discrete 0.053s (19.000 fps)
        Interval: Discrete 0.056s (18.000 fps)
        Interval: Discrete 0.062s (16.000 fps)
        Interval: Discrete 0.067s (15.000 fps)
```

確認相機設備節點與規格

```
v4l2-ctl -d2 --list-formats-ext | grep Size
```

(用 `| grep Size` 篩選出包含 "Size" 關鍵字之行)

```
root@tda4vm-sk:/opt/edgeai-gst-apps# v4l2-ctl -d2 --list-formats-ext | grep Size
[ 997.641270] usb 1-1.1: reset high-speed USB device number 3 using xhci-hcd
Size: Discrete 640x480
Size: Discrete 160x120
Size: Discrete 176x144
Size: Discrete 320x176
Size: Discrete 320x240
Size: Discrete 352x288
Size: Discrete 432x240
Size: Discrete 544x288
Size: Discrete 640x360
Size: Discrete 752x416
Size: Discrete 800x448
Size: Discrete 800x600
Size: Discrete 864x480
Size: Discrete 960x544
Size: Discrete 960x720
Size: Discrete 1024x576
Size: Discrete 1184x656
Size: Discrete 1280x720
Size: Discrete 1280x960
Size: Discrete 640x480
Size: Discrete 160x120
Size: Discrete 176x144
Size: Discrete 320x176
Size: Discrete 320x240
Size: Discrete 352x288
Size: Discrete 432x240
Size: Discrete 544x288
Size: Discrete 640x360
Size: Discrete 752x416
```


物件辨識配置檔

物件辨識範例設定

- ❖ 查看內建配置檔(.yaml)
- ❖ 在configs資料夾中可以查看所有內建配置檔
- ❖ 本次將使用object_detection.yaml實作

```
cd ./configs
```

```
ls
```

```
root@tda4vm-sk:/opt/edgeai-gst-apps# ls
CONTRIBUTING  README.md  apps_cpp  configs  download_models.sh  init_script.sh  log.txt  scripts  tests
LICENSE        apps-python  apps_python  docker  download_test_data.sh  log.log  optiflow  setup_script.sh

root@tda4vm-sk:/opt/edgeai-gst-apps# cd ./configs
root@tda4vm-sk:/opt/edgeai-gst-apps/configs# ls
app_config_template.yaml  http_src_example.yaml  object_detection.yaml  remote_display.yaml  shared_model_instance.yaml
display_and_encode_example.yaml  image_classification.yaml  object_detection2.yaml  rpiV2_cam_example.yaml  single_input_multi_infer.yaml
face_detection.yaml  imx390_cam_example.yaml  od_with_two_sink.yaml  rtsp_src_example.yaml
```

物件辨識範例設定

vi object_detection.yaml

- ❖ 使用 vi 編輯 object_detection.yaml
- ❖ 將更改輸入資料、輸出目的地、推論用模型

```
root@tda4vm-sk:/opt/edgeai-gst-apps/configs# vi object_detection.yaml
```

開啟檔案後畫面



```
title: "Object Detection"
log_level: 2
inputs:
  input0:
    source: /dev/video2
    format: jpeg
    width: 1280
    height: 720
    framerate: 30
  input1:
    source: /opt/edgeai-test-data/videos/video0_1280_768.h264
    format: h264
    width: 1280
    height: 768
    framerate: 30
    loop: True
  input2:
    source: /opt/edgeai-test-data/images/%04d.jpg
    width: 1280
    height: 720
    index: 0
    framerate: 1
    loop: True
models:
  model0:
    model_path: /opt/model_zoo/TVM-0D-5120-ssdLite-mobDet-DSP-coco-320x320
    viz_threshold: 0.6
  model1:
    model_path: /opt/model_zoo/TFL-0D-2020-ssdLite-mobDet-DSP-coco-320x320
    viz_threshold: 0.6
  model2:
    model_path: /opt/model_zoo/ONR-0D-8220-yolox-s-lite-mmdet-coco-640x640
    viz_threshold: 0.6
```

物件辨識範例設定

❖ 編輯inputs部分更改輸入資料設定

- 每個子標籤 (input<數字>) 部分都是輸入資料設定的名稱，source代表輸入資料來源，其他為尺寸、格式設定
- 可以自由新增input，來源可以是影片、圖片、相機等等

e.g: input0、input1...

```
title: "Object Detection"
log_level: 2
inputs:
  input0:
    source: /dev/video2
    format: jpeg
    width: 1280
    height: 720
    framerate: 30
  input1:
    source: /opt/edgeai-test-data/videos/video0_1280_768.h264
    format: h264
    width: 1280
    height: 768
    framerate: 30
    loop: True
  input2:
    source: /opt/edgeai-test-data/images/%04d.jpg
    width: 1280
    height: 720
    index: 0
    framerate: 1
    loop: True
```

物件辨識範例設定

❖ 編輯models部分更改推論模型設定

- 每個子標籤 (model<數字>) 都是推論模型設定的名稱，
model_path代表推論模型來源資料夾
- 可用的models 都在/opt/model_zoo內
- 也可以自由新增model

```
models:
  model0:
    model_path: /opt/model_zoo/TVM-OD-5120-ssdLite-mobDet-DSP-coco-320x320
    viz_threshold: 0.6
  model1:
    model_path: /opt/model_zoo/TFL-OD-2020-ssdLite-mobDet-DSP-coco-320x320
    viz_threshold: 0.6
  model2:
    model_path: /opt/model_zoo/ONR-OD-8220-yoloX-s-lite-mmdet-coco-640x640
    viz_threshold: 0.6
```

物件辨識範例設定

❖ 編輯outputs部分更改輸出目的地設定

- 每個子標籤 (output<數字>) 都是輸出目的地設定的名稱
- sink代表輸出目的地名稱 (如果是路徑就是輸出檔案)
- kmssink為螢幕輸出

```
outputs:
  output0:
    sink: kmssink
    width: 1920
    height: 1080
    overlay-perf-type: graph
  output1:
    sink: /opt/edgeai-test-data/output/output_video0.mkv
    width: 1920
    height: 1080
  output2:
    sink: /opt/edgeai-test-data/output/output_image_%04d.jpg
    width: 1920
    height: 1080
  output3:
    sink: remote
    width: 1920
    height: 1080
    port: 8081
    host: 127.0.0.1
    encoding: jpeg
    overlay-perf-type: graph
```


物件辨識範例設定

❖ 編輯 flow 部分更改資料流設定

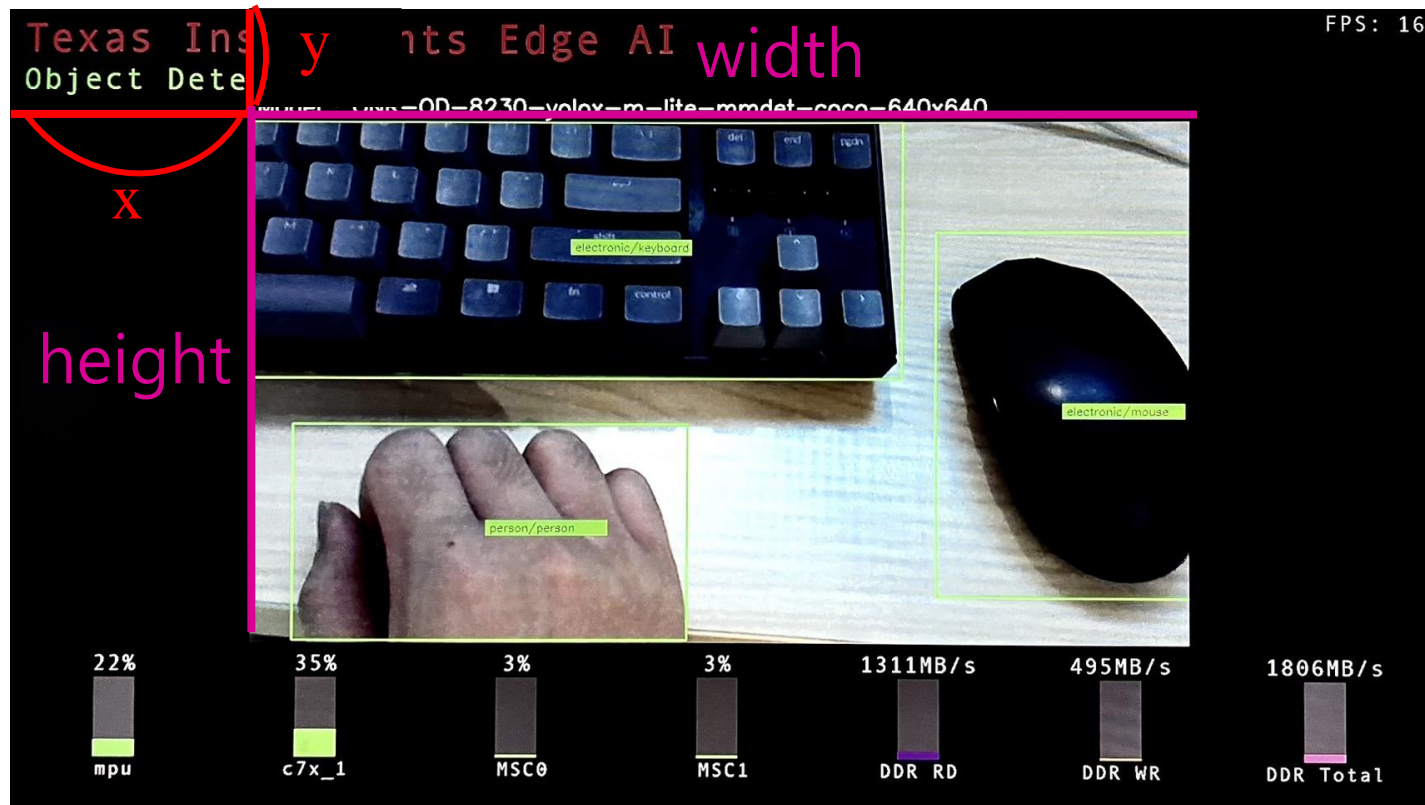
- 用來設定資料該從何來、用什麼模型、輸出到哪
- 可以定義多個 flow 來實現多輸入、同時多模型多推理
- 在中括號 ([]) 內依序打上
 - 輸入資料設定名稱、推論模型設定名稱、輸出目的地設定名稱、顯示準位[offset(x), offset(y), 輸入尺寸width, 輸入尺寸height]
 - 需注意尺寸勿超出輸出尺寸(以螢幕為例)
 - $320+1280 = 1600 (<1920)$ 、 $150+720 = 870 (<1080)$

```
flows:  
flow0: [input0,model2,output0,[320,150,1280,720]]
```

物件辨識範例設定

補充:

顯示準位[offset(x), offset(y), 輸入尺寸width, 輸入尺寸height]



實測切換不同模型之效果

實測切換不同模型之效果

查看內建模型

在model_zoo中可以查看內建可使用模型，本次將會比較以下三種模型的**準確率**和**C71x的使用率**

- ONR-OD-8220-yolox-s-lite-mmdet-coco-640x640
- ONR-OD-8270-yolox-pico-lite-mmdet-coco-320x320
- ONR-OD-8200-yolox-nano-lite-mmdet-coco-416x416

```
root@tda4vm-sk:/opt/edgeai-gst-apps# cd ..
root@tda4vm-sk:/opt# ls
containerd          edgeai-gst-apps      edgeai-test-data    imaging             oob-demo-assets    tidl_test
edgeai-apps-utils  edgeai-gst-plugins  edgeai-tiovx-kernels ltp                 ti-gpio-cpp         vision_apps
edgeai-dl-inferer  edgeai-studio-agent edgeai-tiovx-modules model_zoo            ti-gpio-py          vx_app_arm_remote_log.out
```

```
root@tda4vm-sk:/opt# cd ./model_zoo/
root@tda4vm-sk:/opt/model_zoo# ls
ONR-CL-6360-regNetx-200mf      ONR-SS-8818-deeplabv3lite-mobv2-qat-robokit-768x432  unsupported_models.txt
ONR-OD-8020-ssd-lite-mobv2-mmdet-coco-512x512          TFL-CL-0000-mobileNetV1-mlperf                       yolov5_640
ONR-OD-8200-yolox-nano-lite-mmdet-coco-416x416         TFL-OD-2020-ssdLite-mobDet-DSP-coco-320x320            yolox
ONR-OD-8220-yolox-s-lite-mmdet-coco-640x640            TFL-SS-2580-deeplabv3_mobv2-ade20k32-mlperf-512x512   yolox_old
ONR-OD-8270-yolox-pico-lite-mmdet-coco-320x320         TVM-CL-3090-mobileNetV2-tv                             yoloxs
ONR-OD-8420-yolox-s-lite-mmdet-widerface-640x640      TVM-OD-5120-ssdLite-mobDet-DSP-coco-320x320
ONR-SS-8610-deeplabv3lite-mobv2-ade20k32-512x512      TVM-SS-5710-deeplabv3lite-mobv2-cocoseg21-512x512
```

實測切換不同模型之效果

用於物件辨識的模型比較

說明：AP(準確率)

參考網站：

model_zoo for **Object Detection**

<https://github.com/TexasInstruments/edgeai-modelzoo/tree/main/models/vision/detection>

Dataset	Model Name	Input Size	GigaMACS	AP[0.5:0.95]%, AP50%	Available	Notes
-	YOLOX models					
COCO	YOLOX-m-Lite	640x640	36.9	44.4, 62.9	Y	
COCO	YOLOX-s-Lite	640x640	13.43	38.3, 56.9	Y	
COCO	YOLOX-tiny-Lite	416x416	3.240	30.5, 47.4	Y	
COCO	YOLOX-nano-Lite (depthwise=False)	416x416	1.476	24.8, 40.1	Y	
COCO	YOLOX-pico-Lite (depthwise=False)	320x320	0.503	17.9, 29.4	Y	
COCO	YOLOX-femto-Lite (depthwise=False)	320x320	0.238	12.7, 21.9	Y	

實測切換不同模型之效果

補充：下載模型(需連上網)可不作

`./download_models.sh`

說明：使用Model Downloader Tool，下載模型

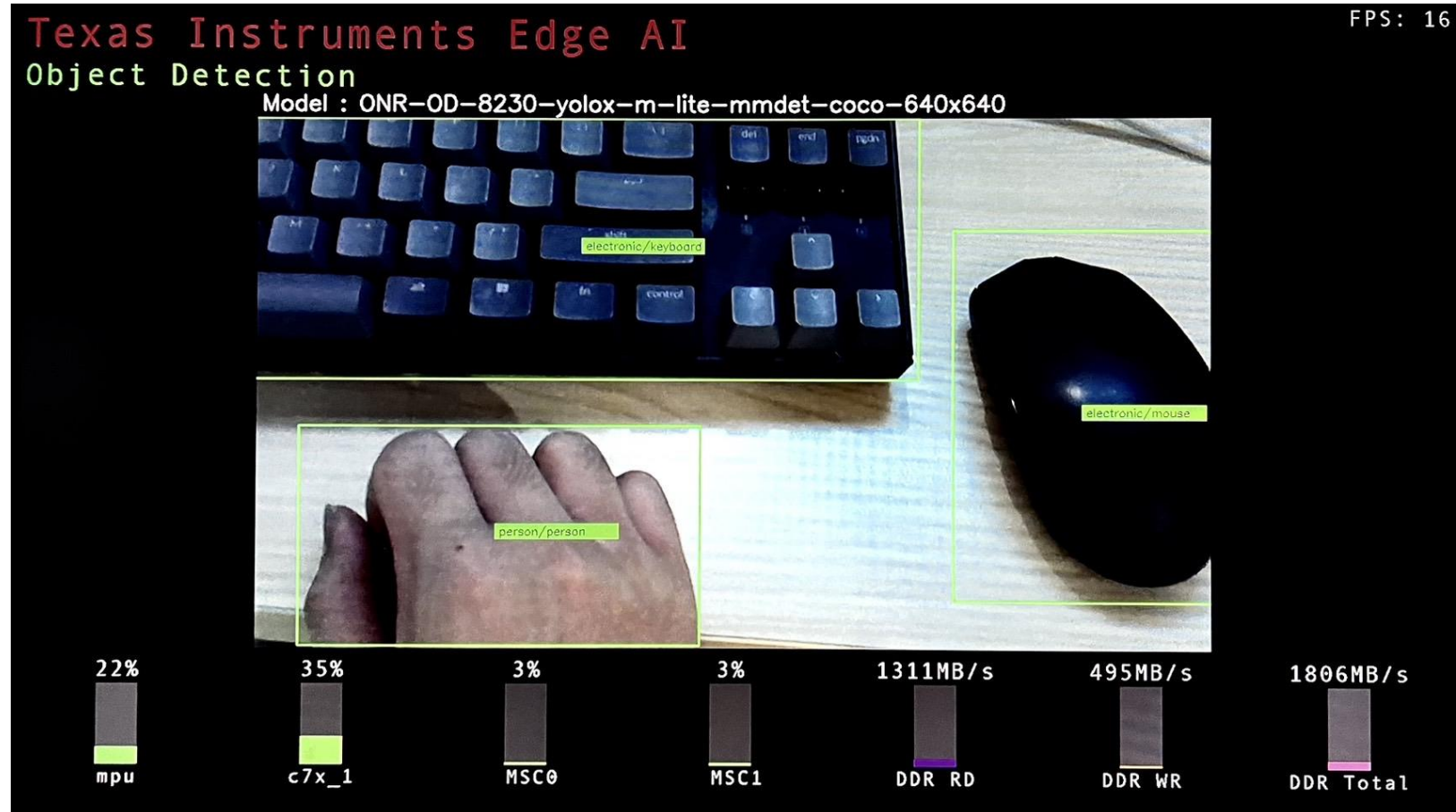
```
root@tda4vm-sk:/opt/edgeai-gst-apps# ./download_models.sh
```

```
Select Models to Download
-----
Model Downloader
-----
Keys:
Up-Down to Navigate Menu
Space to Select Models
Enter to Continue

[ ] classification all models
[ ]   ONR-CL-6090-mobileNetV2-tv      14M
[ ]   ONR-CL-6098-mobileNetV2-tv-qat  14M
[ ]   ONR-CL-6100-resNet18            48M
[ ]   ONR-CL-6110-resNet50           104M
[ ]   ONR-CL-6160-regNetX-400mf-tv    23M
[ ]   ONR-CL-6170-regNetX-800mf-tv    30M
[ ]   ONR-CL-6180-regNetX-1.6gf-tv    38M
[*]   ONR-CL-6360-regNetx-200mf      11M
[ ]   ONR-CL-6480-mobv3-lite-small    8M
[ ]   ONR-CL-6488-mobv3-lite-small-qat 8M
[ ]   ONR-CL-6490-mobv3-lite-large   16M
[*]   TFL-CL-0000-mobileNetV1-mlperf  17M
[ ]   TFL-CL-0010-mobileNetV2       14M
[ ]   TFL-CL-0080-mobileNet-edgeTPU-mlperf 17M
[ ]   TFL-CL-0090-efficientNet-edgeTPU-s 23M
[ ]   TFL-CL-0130-efficientNet-lite0  19M
[ ]   TFL-CL-0160-resNet50V1p5-mlperf 106M
[*]   TVM-CL-3090-mobileNetV2-tv     43M
[ ] detection all models
[*]   ONR-OD-8020-ssd-lite-mobv2-mmdet-coco-512x512 13M
-----
v(+)-50%
-----
< OK > < Quit >
```


實測切換不同模型之效果

相機輸入實測範例(單輸入單推理)

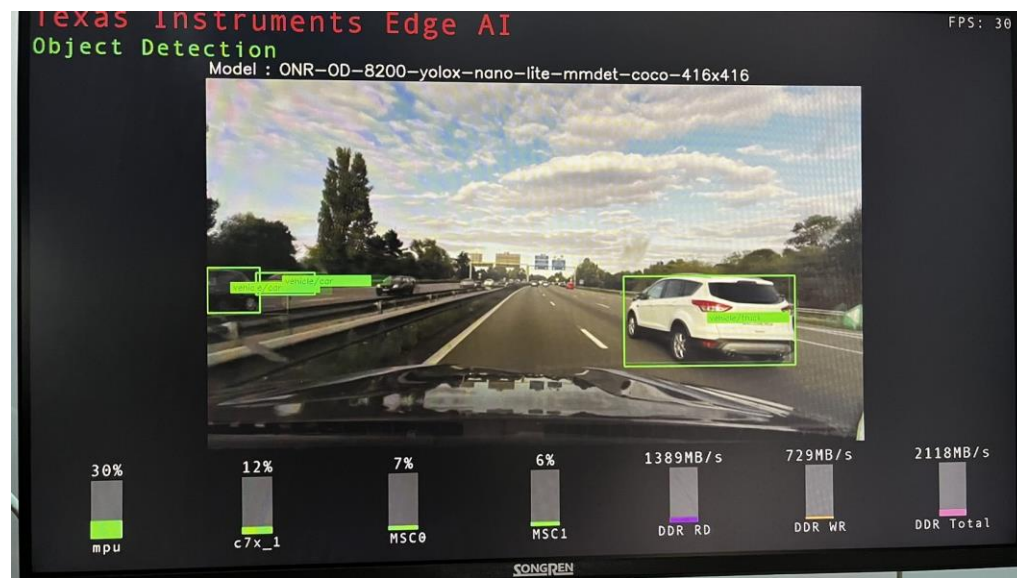


實測切換不同模型之效果

影片輸入比較模型實測範例: C71x使用率、AP



Pico-lite



nano-lite

實測切換不同模型之效果

影片輸入比較模型實測範例: C71x使用率、AP



s-lite



m-lite

實測切換不同模型之效果

相機、影片輸入實測範例(多輸入多推理)

影片源:

1. Video0_1280_760.h264
2. oob-gui-video9.h264

注意:

1. Format 為 h264 或 auto
2. 輸出尺寸勿超出螢幕尺寸
3. 共需要四個flow

實驗目標

- ❖ 完成整體Lab3實驗各模型比較lab2之準確和DSP資源使用率紀錄

參考資料與文獻

- [1] [TDA4VM Processors datasheet \(Rev. K\)](#)
- [2] [J721E DRA829/TDA4VM Processors Silicon Revision 1.1/1.0 \(Rev. D\)](#)
- [3] [DRA829/TDA4VM Technical Reference Manual \(Rev. C\)](#)
- [4] [Jacinto7 AM6x, TDA4x, and DRA8x High-Speed Interface Design Guidelines \(Rev. A\)](#)
- [5] [TMS320C6652 and TMS320C6654 Fixed and Floating-Point Digital Signal Processor datasheet \(Rev. E\)](#)
- [6] [TMS320C6652/54/55/57 Multicore Fixed and Floating-Point DSP SR1.0 \(Rev. C\)](#)
- [7] [SK-TDA4VM User's Guide \(Rev. D\)](#)
- [8] [J721EXSKG01EVM EU Declaration of Conformity \(DoC\) \(Rev. A\)](#)
- [9] [DMA Controller Module \(Chapter Excerpt From MSP430x5xx Family, SLAU208\) \(Rev. F\)](#)
- [10] [AM437x Sitara™ Processors datasheet \(Rev. E\)](#)
- [11] [AM437x and AMIC120 ARM® Cortex™-A9 Processors Technical Reference Manual \(Rev. I\)](#)
- [12] [TMS320C6000 DSP Cache User's Guide \(Rev. A\)](#)