Bringing Video Analytics from the PC to the DSP

Gary Myers
VP, Engineering
ObjectVideo

Upendra Patel
CTO
eInfoChips, Ltd
Topics

• ObjectVideo Overview
• Video Analytics Overview
• Moving to DM642
• Moving to DaVinci
• eInfoChips DaVinci IP Camera Demo
• Q&A
ObjectVideo Overview

- **Company background**
  - Founded in 1998 by DoD R&D managers
  - Product developed in collaboration with the US DoD (DARPA)
  - Continued R&D and collaborative development with U.S. government
  - 100+ employees
  - Based in Reston, VA – Offices in CA, UK and Spain

- **The leader in IVS**
  - ObjectVideo is the current market leader at over 31% share
  - Intel and TI platforms
  - 175+ end user customers in 235+ global installations
  - Growing OEM business – over 685,000 TI-DSP based licenses committed and growing

- **ObjectVideo software turns video into manageable data**
  - Current software applications span from the Homeland to the home
  - ObjectVideo software makes the entire video ecosystem smart
What We Do

• ObjectVideo software empowers video systems to become intelligent

• By turning video into data, devices can respond and alert on a variety of human and non-human events, such as:
  – Classifications of people and vehicles
  – A person or vehicle breaking a perimeter
  – A person leaving a bag unattended
  – A person or vehicle loitering beyond a user determined amount of time
  – An object being stolen

• Video intelligence should exist throughout the entire video eco-system including cameras, encoders, routers, storage, etc.
  – Intelligence at the edge
  – Smart routing
  – Optimal bandwidth management
  – Efficient recording
  – Intelligent video archiving and retrieval
ObjectVideo Example – Tripwire

1. Add Rule
2. Activate Rule
3. Receive Alert

“Detect when <Human> crosses <Alert Perimeter> - send alert via <E-mail> and <Alert Console>”
How ObjectVideo Works

"Watches" Video

Analyzes Video

Separates Objects from Environment

Describes Objects and Activities (metadata)

Meta Data

Activity Detection

- Shape
- Size
- Relative Velocity
- Trajectory
- Time Present
- Classification
- (other)

Video Feed

ObjectVideo ONBOARD

ObjectVideo FORENSICS

Technology for Innovators™

TI Developer Conference
ObjectVideo Product Line

Initial Release
January 2003

Forensics Introduced
April 2004

OnBoard Introduced
May 2005

OnBoard DaVinci
March 2007
Moving to TI’s DM642

• Reasons for migration from PC
  – Form factor
  – Price point

• Reasons for choosing the DM642
  – TI Market Leader in DSPs
  – DM642 is a common platform for video encoders
What’s OnBoard?

“Watches” Video

Activity Inference

- Shape
- Size
- Relative Velocity
- Trajectory
- Time Present
- Classification
- (other)

Separates Objects from Environment

Describes Objects and Activities (metadata)

Meta Data

Content Analysis

Video Feed

ObjectVideo

FORENSICS

ObjectVideo ONBOARD

INTELLIGENT VIDEO

Minds in Motion

Technology for Innovators™

TI Developer Conference
Moving to DM642: Process

1. C++/C# Code
   - Port to C
     - C Code
       - C Code Output Matches?
         - Yes: Port to DSP
         - No: Original Test Output

2. Optimized C Code
   - Optimized Code Output Matches?
     - Yes: Optimize for DSP
     - No: DSP Code Output Matches?
Moving to DM642: C Porting

- Port from C++/C#
  - Inheritance
  - Polymorphism
  - Rich libraries
    - Collection classes

- Memory Utilization
  - Reduce/restrict memory allocations
    - PC code allocated memory whenever needed
    - DSP code restricts memory allocation to startup
  - Finite resources
Moving to DM642: DSP Porting

• Compile for the DSP
• Utilize appropriate system calls
  – Create abstraction layers for platform calls
• Address alignment
  – Make addresses 4-byte aligned
    • If not, read/write through a pointer will be incorrect
• Data type conversion
Moving to DM642: Optimization

- Float to fixed conversions
  - Find appropriate Q Format (e.g. Q20)
    - 20 bits for fractional part
    - 11 bits for integer
    - 1 bit for sign
- #pragma usage
  - MUST_ITERATE, UNROLL, etc.
- Cache usage
  - CACHE_invL2, CACHE_wbL2, etc.
- DMA access
- Parallel execution
- Intrinsic C usage
  - _ssub, _sadd, etc.
- ISRAM utilization for quicker access times
Float to Fixed Example

- Macro to convert float to fixed
  ```c
  #define fp2fixed(flt, q) ((int)((flt)*(1<<(q))))
  ```
- Usage:
  ```c
  int fixed = fp2fixed(3.56435, 20);
  ```
- Operations
  - Add/subtract with same q; same as int
  - Multiply/Divide more complicated
    - 1.25 * 1.25 = 1.5625 (double the fractional part)
    - Requires more bits so more truncation can occur (32bit*32bit = 64bit value)
Comparison

**VEW**
Memory Utilization: ~250MB  
CPU Utilization: ~1Ghz @ 320x240  
Framerate: 15+fps  
(Pentium Xeon)

**OnBoard**
Memory Utilization: 16-24MB  
CPU Utilization: <180Mhz @320x240  
Framerate: 10fps  
(DM642 600Mhz)
Moving to DaVinci

• Reasons
  – Reduced BOM
  – Shorter development cycle
  – Interface standards

• Chip selections
  – 6446 (with ARM)
  – 643X (without ARM)
DSP Differences

- 644x/643x supports DDR2
- 644x has reduced L2 cache (64KB)
  - DM642: 256KB
  - DM643x: 64-128KB
- 6446 has slightly reduced clock speed (594Mhz)
  - DM642: 600/720Mhz
  - DM644x/DM643x: Varies with chip
- 6446 has single video port
  - DM642: 3 video ports
DaVinci: Quick Review

High-level 6446 Architecture

ARM
DSP
Vendor Video Encoder/Decoder
Vendor Image Encoder/Decoder
Vendor Speech Encoder/Decoder
Vendor Audio Encoder/Decoder

Main Application
VISA API
CODEC Engine
Video
Image
Speech
Audio
(Stubs)
RPC
RPC
RPC
RPC

CODEC Engine
Video
Image
Speech
Audio
(Skeletons)
DaVinci Review: xDAIS

- **xDAIS**
  - eXpressDSP Algorithm Standard
  - Existing interface to provide a common mechanism to instantiate a DSP algorithm
  - IALG interface
    - Abstracts memory management
    - Standard definition across all algorithms
  - IMOD interfaces
    - Custom extension to the IALG interface
  - IDMA interface
    - Abstracts DMA management
IALG Interface

int (*algNumAlloc)(void);

int (*algAlloc)(const IALG_Params *, struct IALG_Fxns **, IALG_MemRec *);

int (*algInit)(IALG_Handle, const IALG_MemRec *, IALG_Handle, const IALG_Params *);

void (*algActivate)(IALG_Handle);

void (*algMoved)(IALG_Handle, const IALG_MemRec *, IALG_Handle, const IALG_Params *);

void (*algDeactivate)(IALG_Handle);

int (*algFree)(IALG_Handle, IALG_MemRec *);
IMOD Interface

Example:

typedef struct I<mod>_Fxns
{
    IALG_Fxns ialg;
    void (*runModule)(I<mod>_Handle handle,
                    XDAS_Int16 in[],
                    XDAS_Int16 out[]);
} I<mod>_Fxns;

Replace <mod> with name of module.
DaVinci Review: xDM

- Digital Media extensions to the xDAIS interface
- Provides common interfaces to algorithm control and processing
  - Pre-defines specific IMOD interfaces
  - Supports vendor specific extensions
- VISA APIs
  - TI defines 8 APIs in 4 distinct areas
    - Video – IVidEnc/IVidDec
    - Imaging – IImgEnc/IImgDec
    - Speech – ISphEnc/ISphDec
    - Audio – IAudEnc/IAudDec
VISA API: Example

• IVidEncode

  XDAS_Int32 (*process)(IVIDENC_Handle handle, XDM_BufDesc *inBufs,
                      XDM_BufDesc *outBufs, IVIDENC_InArgs *inArgs,
                      IVIDENC_OutArgs *outArgs);

  XDAS_Int32 (*control)(IVIDENC_Handle handle, IVIDENC_Cmd id,
                       IVIDENC_DynamicParams *params, IVIDENC_Status *status);

• IVidDecode

  XDAS_Int32 (*process)(IVIDDEC_Handle handle, XDM_BufDesc *inBufs,
                      XDM_BufDesc *outBufs, IVIDDEC_InArgs *inArgs,
                      IVIDDEC_OutArgs *outArgs);

  XDAS_Int32 (*control)(IVIDDEC_Handle handle, IVIDDEC_Cmd id,
                       IVIDDEC_DynamicParams *params, IVIDDEC_Status *status);
typedef struct IVIDENC_Fxns
{
    IALG_Fxns ialg;

    XDAS_Int32 (*process)(IVIDENC_Handle handle,
                          XDM_BufDesc *inBufs,
                          XDM_BufDesc *outBufs,
                          IVIDENC_InArgs *inargs,
                          IVIDENC_OutArgs *outargs);

    XDAS_Int32 (*control)(IVIDENC_Handle handle,
                          IVIDENC_Cmd id,
                          IVIDENC_DynamicParams *params,
                          IVIDENC_Status *status);

} IVIDENC_Fxns;
DaVinci Review: Codec Engine

- Provides RPC mechanism from ARM to DSP
DaVinci: Adding OnBoard

High-level 6446 Architecture
Steps to DaVinci

- xDAIS compliance
- Define xDM interface
- Codec Engine support
- DaVinci-specific optimizations
- Testing
xDAIS Compliance

• Add support for IALG interface
• Utilize DMAN3/ACPY3 throughout the code
Define xDM Interface

• VISA APIs does not currently have an exact match for Video Analytics
  – Current APIs are encoder/decoder focused
• New top-level API defined
  – IVidAnalytics
  – Follows similar format as existing APIs but Video Analytics focused
    • Existing APIs were focused on encode/decode operations with an binary data stream output for every frame
  – Joint effort between OV and TI
• OV-specific Extension
  – IOVAnalytics
typedef struct IVIDENC_Params {
    XDAS_Int32 size;                 /* size of this structure */
    XDAS_Int32 encodingPreset;       /* Encoding preset */
    XDAS_Int32 rateControlPreset;    /* Rate control presets */
    XDAS_Int32 maxHeight;            /* Maximum video height to be supported */
    XDAS_Int32 maxWidth;             /* Maximum video width to be supported */
    XDAS_Int32 maxFrameRate;         /* Maximum Framerate * 1000 to be supported */
    XDAS_Int32 maxBitRate;           /* Maximum Bitrate to be supported in bits per second */
    XDAS_Int32 dataEndianness;       /* Endianness of output data. Refer to video data format enum above */
    XDAS_Int32 maxInterFrameInterval; /* I to P frame distance. e.g. = 1 if no B frames, 2 to insert one B frame */
    XDAS_Int32 inputChromaFormat;    /* Set to XDM_ChromaFormat type. */
    /* Set to DEFAULT to avoid re-sampling. */
    XDAS_Int32 inputContentType;     /* XDM Content Type: IVIDEO_PROGRESSIVE, IVIDEO_INTERLACED */
} IVIDENC_Params;
typedef struct IVIDANALYTICS_Params {
    XDAS_Int32 size;             /* size of this structure */
    XDAS_Int32 maxHeight;        /* Maximum video height to be supported */
    XDAS_Int32 maxWidth;         /* Maximum video width to be supported */
    XDAS_Int32 maxFrameRate;     /* Maximum Framerate * 1000 to be supported */
    XDAS_Int32 dataEndianness;   /* Endianness of output data. Refer to video data format enum above */
    XDAS_Int32 inputChromaFormat; /* Set to XDM_ChromaFormat type.
                                     * Set 0 for FRAME FORMAT YUV 4:2:0
                                     * Set 1 for FRAME FORMAT YUV 4:2:2 */
    XDAS_Int32 maxViews;         /* Maximum views available for analysis */
    XDAS_Int32 maxRulesPerView;  /* Maximum rules allowed per view */
} IVIDANALYTICS_Params;
IMOD Definition for IVIDANALYTICS

typedef struct IVIDANALYTICS_Fxns
{
    IALG_Fxns ialg;

    XDAS_Int32 (*process)(IVIDANALYTICS_Handle handle,
                           XDM_BufDesc *inBufs,
                           XDM_BufDesc *outBufs,
                           IVIDANALYTICS_InArgs *inargs,
                           IVIDANALYTICS_OutArgs *outargs);

    XDAS_Int32 (*control)(IVIDANALYTICS_Handle handle,
                           IVIDANALYTICS_Cmd id,
                           IVIDANALYTICS_DynamicParams *params,
                           IVIDANALYTICS_Status *status);

} IVIDANALYTICS_Fxns;
Example Base Structures

typedef struct IVIDANALYTICS_InArgs
{
    XDAS_Int32 size;    /* size of this structure */
} IVIDANALYTICS_InArgs;

typedef struct IVIDANALYTICS_Status
{
    XDAS_Int32 size;    /* size of this structure */
    XDAS_Int32 extendedError;    /* Extended Error code. */
    XDM_AlgBufInfo bufInfo;     /* Input & output buffer information */
    XDAS_Int32 viewState;      /* View status of the Analytics Engine */
    XDM_BufDesc *dynamicStatusBufs;    /* Multi-purpose status buffers */
} IVIDANALYTICS_Status;
Extending Structures

typedef struct IOVANALYTICS_InArgs
{
  IVIDANALYTICS_InArgs vidanalyticsInArgs; /* Base structure reference */

  /* OV Specific members */
  XDAS_UInt8 absoluteTime[8]; /* Application Time - 8 byte */
  XDAS_UInt32 relativeTimeInMillis; /* Time difference in ms */
  ...
  XDAS_Bool flushData; /* Signals a flush operation */
}
} IOVANALYTICS_InArgs;

typedef struct IOVANALYTICS_Status
{
  IVIDANALYTICS_Status vidanalyticsStatus; /* Base structure reference */

  /* OV specific status here */
  XDAS_UInt8 version[4]; /* Version of library */
  IOVANALYTICS_UserActions userActions; /* Flags indicating user actions */
  ...
  XDAS_Bool isActivated; /* TRUE if the sensor is activated */
}
} IOVANALYTICS_Status;
Codec Engine Support

• Stubs and skeletons

• Build and package OVAnalytics Library
  – Build DSP library
    • Placed under ./analytics/ovanalytics/lib directory
  – Build stubs and skeletons
    • Placed under ./extensions/ovanalytics directory
Creating a Stub

static XDAS_Int32 process (IOVANALYTICS_Handle h, XDM_BufDesc *inBuf, XDM_BufDesc *outBuf, IOVANALYTICS_InArgs *inArgs,
   IOVANALYTICS_OutArgs *outArgs)
{
    ...
    VISA_Handle visa = (VISA_Handle)h;
    _OVANALYTICS_Msg *msg;
    IOVANALYTICS_Event *event;
    /* get a message appropriate for this algorithm */
    if ((msg = (_OVANALYTICS_Msg *)VISA_allocMsg(visa)) == NULL)
    {
        return (OVANALYTICS_ERUNTIME);
    }
    /* Specify the processing command that the skeleton should do */
    msg->visa.cmd = _OVANALYTICS_CPROCESS;
    ...
    /* inBuf is a pointer, so we have to convert it */
    inBuf->bufSizes = (XDAS_Int32*) Memory_getBufferPhysicalAddress(inBuf->bufSizes, sizeof (XDAS_Int32)*inBuf->numBufs, NULL);
    inBuf->bufs = (XDAS_Int8**) Memory_getBufferPhysicalAddress (inBuf->bufs, sizeof (XDAS_Int8 *) * inBuf->numBufs, NULL);
    ...
    inBuf = (XDM_BufDesc *) Memory_getBufferPhysicalAddress (inBuf, sizeof (XDM_BufDesc), NULL);
    ...
    /* send the message to the skeleton and wait for completion */
    retVal = VISA_call(visa, (VISA_Msg *)&msg);
    /* inBuf is a pointer, so we have to convert back to virtual from physical */
    inBuf = (XDM_BufDesc *)Memory_getBufferVirtualAddress ((ULInt32)inBuf, sizeof(XDM_BufDesc));
    ...
    VISA_freeMsg (visa, (VISA_Msg)msg);
    return retVal);
}
Creating a Skeleton

static VISA_Status call (VISA_Handle visaHandle, VISA_Msg visaMsg)
{
    _OVANALYTICS_Msg *msg = (_OVANALYTICS_Msg *)visaMsg;
    OVANALYTICS_Handle handle = (OVANALYTICS_Handle)visaHandle;
    ... /* perform the requested operation by parsing message. */
    switch (msg->visa.cmd)
    {
    case _OVANALYTICS_CPROCESS:
        {
            /* invalidate cache for input and output buffers */
            if (inBuf != NULL) Memory_cacheInv (inBuf, sizeof (XDM_BufDesc));
            ...
            /* make the process call */
            msg->visa.status = OVANALYTICS_process (handle, inBuf, outBuf, &(msg->cmd.process.inArgs), &(msg->cmd.process.outArgs));
            ...
            /* flush cache for output buffer */
            ...
            if (outBuf != NULL)
            Memory_cacheWblInv (outBuf, sizeof (XDM_BufDesc));
            break;
        }
    case _OVANALYTICS_CCONTROL:
        {
            ...
            break;
        }
    ... break;
    } return (VISA_EOK); }

Utilizing OV Analytics

• Build application
  – Build DSP server containing all necessary algorithms
  – Build ARM application
• Run application
  – Start ARM application
    • Automatically loads DSP server
Implementation Notes

• 32-bit structure alignment
• Shared memory between ARM and DSP
• Stubs/Skeletons for IALG functions are not written but provided as part of Codec Engine
  – No passing of pointers to any IALG functions
eInfochips Smart Surveillance System Reference Design
eInfochips - Introduction

• IP-Leveraged Electronic Systems Design Services Company
  – Embedded systems design
  – Chip design and verification
• Premier TI Third Party Network Member
  – Authorized Software Provider
  – 5 years of Proven track record
  – Application domains: Streaming Media, Surveillance, Medical Imaging, Automotive Infotainment
  – Platform Expertise: C6000, DM64x, DM320, DM6446
  – Product design services: Conceptual Design to Volume Manufacturing
Smart Surveillance Camera Reference Design
“Ready-to-Go” Solution

- Web Viewer Application with GUI
- H.264, MPEG4, JPEG, G.7xx
- Object Video Content Analytics
- Codec Engine DSP/BIOS™ Link DSP/BIOS™
- MontaVista Linux
- Firmware & Drivers
- Production Ready Hardware

- Rapid market entry
- Optimized BOM
- eInfochips Support Team

Technology for Innovators™
Smart Camera: Hardware Design

- BOM Reduced by $40
- Development Cycle Shortened by 4 Months
IP NetCam Software Framework

APL
- Command Control
- System Management
- Audio Video Streaming
- Connectivity
- Video Analytics App
- Storage
- Manufacturing Diagnostic
- User Diagnostic

EPSI API
Input Output Layer (IOL)
- Video
- Audio
- DSP/Link
- NAND
- MSP 430
- EMAC
- MMC/SD
- 802.11
- PWM
- 2W Dogs
- GPIO

Signal Processing Layer (SPL)
Codec Engine Resource Server
- xDM MPEG4
- xDM H.264
- xDM JPEG
- xDM G.711
- xDM OV

Part of DVEVM
Modified/Developed/Ported by eInfochips
ObjectVideo IP
TI’s Future Release

Technology for Innovators™
Feature-rich Application

- Camera configuration: IP address, selection of encoding formats, resolution etc.
- Camera Control: Imaging parameters, PTZ
- Security Features: Defining rules & customizing alerts
IP Netcam Platform Addresses Key Market Requirements

Complete Hardware and Software Flexibility to Create Optimized Netcams
- Completely flexible scalable development of hardware & software features
- Portability of codecs/algorithms with XDM compliant architecture

Best in Class Features & Performance with DaVinci
- Market-proven video analytics from ObjectVideo on DaVinci
- Separate stream for remote monitor and network storage

Rapid Proof-of-Concept and Customized Product Development
- Complete Sensor to Storage hardware
- Integrated suite of software development framework & tools

Save > 15 Person Months of Development Time
**TI Developer Conference**

Complete Hardware & Software Flexibility to create Smart IP Netcams

<table>
<thead>
<tr>
<th>Hardware</th>
</tr>
</thead>
<tbody>
<tr>
<td>BOM optimized Camera form factor board</td>
</tr>
<tr>
<td>Supports CMOS/ CCD Sensors</td>
</tr>
<tr>
<td>Multiple storage &amp; connectivity options</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Software</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plug-n-play codecs – MPEG4, H.264, MJPEG, customer-proprietary</td>
</tr>
<tr>
<td>Mix-n-match Onboard with codecs for dual streaming OR</td>
</tr>
<tr>
<td>Choose triple stream with 2 MPEG-4 streams &amp; MJPEG for local SD storage</td>
</tr>
<tr>
<td>Standard application framework as SDK</td>
</tr>
</tbody>
</table>

**eInfochips’ Customization Support**

- Access to eInfochips’ team of DaVinci experts
- Web based GUI for configuration & administration
- Customization and manufacturing support
- Integration / porting service for existing software

**Build Range of Cameras**

**Easily & Quickly**

**Minds in Motion**

**Technology for Innovators™**
Surveillance Features

- **Video Input**
  - Micron 3 Megapixel, 1/2” Progressive Scan CMOS Sensor
    - Electronic PTZ control
  - Support for Auto White Balance, Day/Night, Lens Shading, Image Scaling features

- **Video Compression:**
  - MPEG4, JPEG and H.264 (User configurable combinations)
  - Resolution Supported: CIF, QCIF, VGA & D1
  - Frame Rate: 30 FPS for all resolutions

- **Content Analysis and Event Alerts/ Alarms**
  - ObjectVideo algorithm for video content analysis
  - User configurable application for generating Alarms and Email notifications
Software Features

• Feature Rich Web Viewer Application
  – Multi-user access levels with password protection
  – Live streaming, Record and playback features

• Security:
  – 128 Bit AES Encryption

• Protocol Support:
  – IP, HTTP, FTP, SMTP, UPnP, NTP, RTP, RTSP

• Local storage (SD Card):
  – Short duration local recording in case of network failure
  – Even based local recording

• System Integration
  – APIs for user software integration and customization
  – Full software programmability allows feature selectability
Q&A

• Questions?

• Online References:
  – DaVinci
    • http://focus.ti.com/lit/ug/sprue67b/sprue67b.pdf
      (Codec Engine Application Developer’s Guide)
    • http://focus.ti.com/lit/ug/sprued6a/sprued6a.pdf
      (Codec Engine Algorithm Creators Guide)
Bringing Video Analytics from the PC to the DSP

Gary Myers
VP, Engineering
ObjectVideo

Upendra Patel
CTO
eInfoChips, Ltd