TDA2X, A SOC OPTIMIZED FOR ADVANCED DRIVER ASSISTANCE SYSTEMS

Dr. Jagadeesh Sankaran, Senior Member Technical Staff, Texas Instruments Incorporated.
Dr. Nikolic Zoran, Member Group Technical Staff, Texas Instruments Incorporated.

ABSTRACT

TDA2X is an optimized scalable system on chip (SoC) solution from Texas Instruments that spans various application areas of ADAS such as front-camera, surround-view and the emerging area of sensor fusion. It accomplishes this through a focused set of heterogeneous processors, brought together in a scalable architecture with a rich set of integrated peripherals, providing an optimal mix of performance in a low power footprint for Advanced Driver Assistance Systems (ADAS) vision analytics. Computer vision algorithms across the various ADAS application systems have a rich variation and diversity in processing requirements along with the need to run them concurrently within challenging thermal budgets. A heterogeneous architecture with various programmable elements allows system developers to map various portions of the algorithms to the architectures that are best suited for the underlying task allowing maximizing system performance and reducing development time and effort in developing these complex systems. Scalability of the architecture by varying the number of cores and clock speeds of these heterogeneous architectures, allows for scalability in performance and power across low, mid and high end products with one software investment. A critical focus on functional safety across the cores and various memories is particularly essential given the mission critical nature of ADAS applications.

Index Terms— ADAS SoC, Front Camera, Surround View, Sensor Fusion and Programmable Vision Accelerators

1. INTRODUCTION

Advanced driver assistance systems are proliferating in their use across multiple application areas such as front camera surround view and sensor fusion as shown in Figure 1. The TDA2x SoC [1] incorporates a heterogeneous, scalable architecture that includes a mix of TI's fixed and floating-point TMS320C66x digital signal processor (DSP) generation cores, Vision AccelerationPac (EVE), ARM Cortex-A15 MPCore™ and dual-Cortex-M4 processors. The integration of a video accelerator for decoding multiple video streams over an Ethernet AVB network, along with the graphics accelerators (SGX544) for rendering virtual views, enables a 3D viewing experience for surround view applications as shown in Figure 2. The TDA2x SoC, also integrates a host of peripherals including multi-camera interfaces and GigB Ethernet with AVB to enable LVDS-based or Ethernet based surround view systems respectively, displays, graphics engines, multiple external memory interfaces, CAN and others.

The TDA2x SoC includes TI's new Vision AccelerationPac (EVE) is a focused purpose built, fully programmable vision accelerator in a high level language environment, which delivers more than 8x improvement in compute performance for advanced vision analytics than existing ADAS systems at same power levels. The Vision AccelerationPac for this family of products includes multiple embedded vision engines (EVEs) offloading the vision analytics functionality from the application processor while also reducing the power footprint.

This paper will focus on the specific innovations incorporated in EVE and the C66x DSP cores, their use in mapping complicated ADAS algorithms along with the benefits and system entitlement obtained through the use of this heterogeneous compute fabric. It will also review the various safety hooks available that can be used in context of functional safety enabling customers to design systems that meet ASIL levels.

2. COMPUTATIONAL CHALLENGES IN ADAS

Front camera applications include having to run multiple applications such as high-beam assist, lane-keep assist, adaptive cruise control, traffic sign recognition, pedestrian / object detection, and collision avoidance on the same processor often with a stringent power budget < 2.5 W and tight latency requirements. Park assist applications include intelligent 2D and 3D surround view and rear collision warning and detection. The TDA2x SoC is also capable of the fusion of radar and camera sensor data, allowing for a more robust ADAS decision-making process in the automobile.

Further automobiles drive through a variety of outdoor locales, weather conditions, speed limits causing additional complexities ensuring a persistent innovation cycle, with new strategies making programmable platforms a huge advantage, allowing car makers to try out a multitude of strategies while still maintaining aggressive schedules for product release. The increase in resolutions along with an increase in computational requirements to make these algorithms robust causes an exponential increase in processing and memory bandwidth requirements. It also

978-1-4799-2893-4/14/$31.00 ©2014 IEEE

requires computer architects and embedded developers to embrace the power of heterogeneous computing, where each of the heterogeneous elements has a unique strength allowing it to excel on specific types of processing. The TDA2x SoC is developed with this spirit, with advances to an industry leading C6000 DSP architecture along with an introduction of Vision AccelerationPAC with the Embedded Vision Engine (EVE) a fully programmable accelerator purpose built from an extensive profiling of low and mid-level vision kernels.

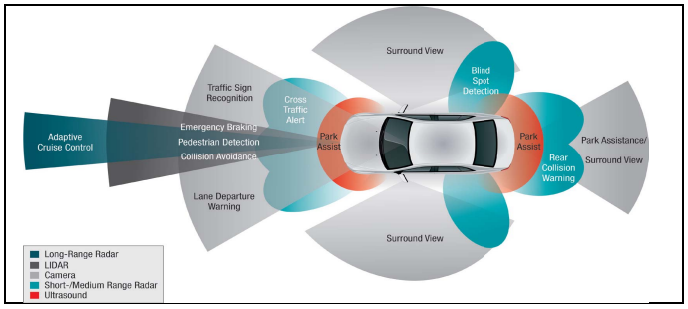

Figure 1: Application Areas for Advanced Driver Assistance Systems (ADAS)

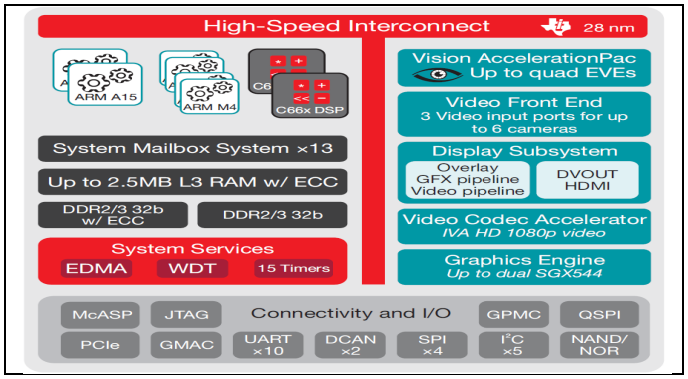

Figure 2: TDA2x SoC ADAS Heterogeneous Architecture

3. ADAS Computer Vision Applications

Computer vision applications, and those in automatic driver assistance systems (ADAS), are comprised of three different types of processing, usually classified as low, mid and high level processing. Most, if not all vision algorithms start off with low level processing, and is characterized by operations on many pixels, characterized by regular accesses borrowing heavily from regular image processing with high computational requirements and memory bandwidth. Mid level vision is the next stage, where we focus on certain objects or regions of interest, that meet a particular criteria for object selection constituting object processing. On these objects, computer vision algorithms choose to evaluate various feature detectors to hand over to high level vision as part of classifier, to eventually classify and track the object as a pedestrian or a traffic sign. Low level vision is best served by wide single instruction working on multiple data (SIMD) architectures to tackle the increasing pixel count, on which the same operation is performed. Mid level vision is best served by multiple instructions working on multiple data (MIMD) to give the flexibility of working on multiple objects in parallel. High level vision includes algorithms with high variability in processing and data access characterized by highly conditional processing such as classifiers and tracking.

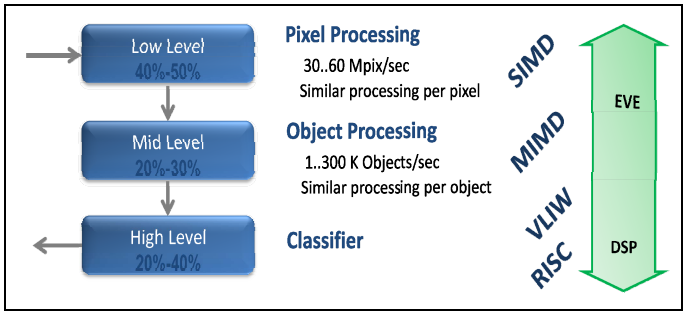

Figure 3: Various Stages of ADAS Computer Vision Applications

4. RELATION TO PRIOR WORK

Current approaches to implementing embedded automotive vision algorithms are truly diverse. They span the gamut of processing platforms, from the use of regular embedded processor cores such as ARM cores, embedded GPU cores, DSP cores, Application Specific Instruction Set Processors (ASIPs) and finally FPGAs. The above classification also illustrates the gradation of the associated programming models used – ranging from standard ANSI-C, C with intrinsic, domain specific languages and finally down to the use of VHDL/Verilog on FPGAs. While the use of ARM cores allows for standard programming models and standard cache based architectures, it also imposes constraints on the levels of performance and energy-efficiency that can be achieved and thus limits the usage of the processor in embedded scenarios. While GPU cores offer a high degree of performance, these high levels of performance are only realizable with large workload sizes with increased result latency and higher power dissipation, often in the range of multiple watts. Lack of support of RTOS or similar infrastructure in GPUs causes challenges in scheduling hard real-time workloads [2].

While FPGAs offer the highest performance and are often used to compensate for any processing blocks that cannot be accomplished by programmable hardware, they face the problem of requiring the algorithm details to be frozen ahead of time, extensive up front engineering efforts, higher costs compared to DSPs, higher power/energy than DSP or ASIP solutions along with bandwidth challenges [3].

5. HETEROGENOUS COMPUTE ARCHITECTURE

A heterogeneous architecture is essential when we need to meet a challenging compute budget with a low power foot

print. The dual core Cortex A15's on TDA2x run at 750 MHZ with an out-of-order superscalar pipeline with a tightly-coupled low-latency level-2 cache with additional improvements in floating point and NEON™. The dual core Cortex M4 is an efficient controller engine for house-keeping tasks including control of IVA-HD encoder and video ports. TI's IVA-HD core is an imaging and video codec accelerator that runs at 532 MHz to enable full HD video encode and decode. Even with all of this compute fire power, vision analytics is still critically dependent on the TMS320C66x DSP for high level vision and Vision AcceleratorPAC EVE for low and mid-level vision to meet the challenging compute and latency needs of advanced driver assistance systems.

5.1. TMS320C66x DSP

TMS320C66x DSP is the most recent DSP core from Texas Instruments and has integrated fixed- and floating-point operation support, leveraging 8-way VLIW to issue up to 8 new operations every cycle, with SIMD operations for fixed point and fully-pipelined instructions, with support for up to 32, 8-bit or 16-bit multiplies per cycle, up to eight, 32-bit multiplies per cycle. Floating point support on the core allows up to eight single-precision multiplies per cycle, or up to two double-precision multiplies per cycle. The C66x DSP is characterized by 128-bits of load-store internal memory bandwidth, with a 32-KB local level-one instruction (L1P) and data (L1D) memories/cache, with a Local level-two unified memory/cache (LL2) which runs at the CPU clock speed, allowing for improved memory bandwidth.

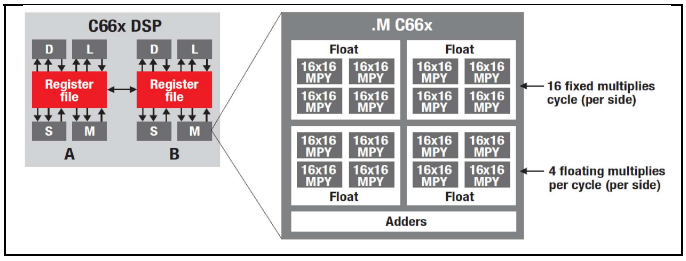

Figure 4: TMS320C66x DSP

5.2. EVE VisionAccelerationPAC

The VisionAccelerationPac contains one or more Embedded Vision Engines (EVE) (Figure 5) at 650 MHZ that deliver programmability, flexibility, low-latency processing as well as power efficiency and a small silicon die area for embedded vision systems. The result is an exceptional combination of performance and value. Each EVE delivers more than 8x improvements in compute performance for advanced vision analytics than existing ADAS systems at same power levels as shown in Figure 6. Within a Vision AccelerationPac is one or more EVE, a vision-optimized processing engine that includes one 32-bit Application-Specific RISC Processor (ARP32) and one 512-bit Vector Coprocessor (VCOP) with built-in mechanisms and unique vision-specialized instructions for concurrent, low-overhead processing. The ARP32 includes 32KB of program cache to enable efficient program execution. It also features a built-in emulation module to simplify debugging and is compatible with TI's Code Composer Studio™ Integrated Development Environment (IDE). There are three parallel flat memory interfaces each with 256-bit load-store memory bandwidth providing a combined 768-bit wide memory bandwidth (6 times higher internal memory bandwidth than most other processors) and has a total of 96KB L1 data memory to enable simultaneous data movement with very low processing latency. Each EVE also has a local dedicated Direct Memory Access (DMA) for data transfer to and from the main processor memory for fast data movement, and a Memory Management Unit (MMU) for address translation and memory protection.

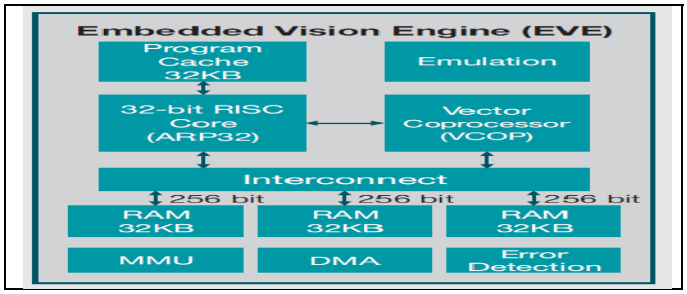

Figure 5: Embedded Vision Engine (EVE)

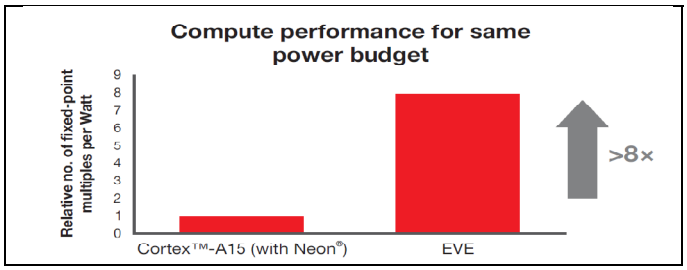

Figure 6: EVE >8× Compute performance for same power budget with respect to A15

To enable reliable operation, each EVE is further equipped with single-bit error detection on all data memories and double-bit error detection on program memory. A key architectural feature is the complete concurrency of the DMA engine, control engine (RISC CPU) and the processing engine (VCOP). This allows, for example, the ARP32 RISC CPU to process an interrupt or execute sequential code, in the meantime the VCOP executes a loop and decodes another in the background, while moving data without any architecture or memory sub-system stalls. It also has built-in support for inter-processor communication by way of hardware mailboxes. EVE enables extreme processing performance and 384-Gbps of internal memory bandwidth, with only 290mW of worst-case total power consumption at 125°C, for a most power-efficient vision processing.

The VCOP vector coprocessor is a Single Instruction Multiple Data (SIMD) engine with built-in loop control and address generation. It provides a dual 8-way SIMD with 16 16-bit multipliers per cycle, for 8-10.4 GMACS per second at 500-650 MHz frequency of sustained throughput, sustained by associated loads/stores and built-in zero-looping overheads with rounding and saturation. It has three-source operations, allowing the two vector units to gain an additional 2x and computes 32 32-bit additions in each cycle. VCOP also has eight address generation units each capable of 4-dimensional address to sustain address for four nested loops and three memory interfaces, resulting in zero overhead for four levels of nested looping. This significantly reduces the compute cycles needed for iterative pixel operations. The vector coprocessor has specialized pipelines for accelerating histogram, weighted histogram and lookup tables and supports common computer-vision processing stages such as gradients, orientation, sorting, bit interleaving/de-interleaving/transposing, integral images and local-binary patterns through specialized instructions. In addition, the vector coprocessor has specialized instructions with flexible and concurrent load store operation to accelerate the region of interest decoding and a scatter/ gather operation for efficient processing of data from non-contiguous memory. This minimizes the conventional data movement and copying needed for traditional image processing procedures resulting in ultra-fast processing performance. Speedups of 4-12x on a diverse range of functions relative to standard processor architectures are common. Sorting is a common computer vision function that occurs in multiple-use cases such as identifying target features to track, and matching in dense optical flow searches. EVE significantly accelerates sorting with custom instruction support, allowing EVE to sort an array of 2048 32-bit data points in < 15.2 μsecs.

6. TDA2X FUNCTIONAL SAFETY

TDA2x SoC is a Quality Managed device with the following salient hooks that can be used in context of functional safety enabling customers to design systems that meet ASIL levels, including SECDED and parity protection in internal memories and external memory interfaces, error detection in EVE and interrupt generation, error handling, timeout with abort mechanism for safe interconnect bus implementation. It also supports illegal instruction aborts and traps on CPUs and CLK routing features to implement clock monitoring, in addition to software-controlled voltage-monitoring options available on-chip. Memory protection units, multiple MMUs and firewalls implement freedom from interference. Multiple clock and power domains are used to avoid common cause failures. Software diagnostics-based approach to detect transient errors that may affect critical system configuration and computation are also available. TDA2x SoC's come with a safety manual as well as detailed FMEDA for customers to assess their safety functions at granular implementation levels and collateral enables seamless implementation of functional safety for customers to achieve ISO 26262 ASIL levels.

7. EVE PROGRAMMING MODEL

The Vision AccelerationPac is fully programmable with a standard set of TI code-generation tools, allowing software to be directly compiled and run on a PC for simulation. The ARP32 RISC core can run full C/C++ programs along with TI's real-time operating system BIOS (RTOS). The VCOP vector coprocessor is programmed with a specialized subset of C/C++ created by TI called VCOP Kernel C. VCOP Kernel C is a templatized C++ vector library which exposes all the capabilities of the underlying hardware through a high-level language. Algorithms written in VCOP Kernel C can be emulated and validated on a standard PC or workstation by using standard compilers such as GNU GCC. This allows developers to incorporate vectorization and verify bit exactness early in the algorithm development process and test with extensive data sets to ensure the robustness of the algorithm. By simply recompiling the source code with TI's code-generation tool, the algorithms can directly run on Vision AccelerationPac. Programs written in VCOP Kernel C have many advantages; they are optimized to leverage the Vision AccelerationPac architecture and instruction set, they have specific loop structure that operates on vector data and they have an almost one-to-one mapping, between C statements and assembly language, resulting in a very efficient code with a small code size and memory footprint. Figure 7, is an example of a vision kernel function that is used often in many settings is a 2-dimensional finite impulse response (FIR) filter.

```
void eve_fir2d
(
    unsigned short blkw, int blkh,
    unsigned short coefw, int coefh,
    int            lofst,
    __vptr_uint8   data_ptr,
    __vptr_uint8   coef_ptr,
    __vptr_uint8   output_ptr,
    int             rnd_bits
 )
{
   __vector V_acc1, Vacc0;
   __vector V_in1,  Vin0;
   __vector V_coef;

   for (I1 = 0; I1 < blkh; I1++)
   {
      for (I2 = 0; I2 < blkw/(2 * VCOP_SIMD_WIDTH); I2++)
      {
         V_acc1 = 0;
         V_acc0 = 0;

         for (I3 = 0; I3 < coefh; I3++)
         {
            for (I4 = 0; I4 < coefw; I4++)
            {
               A0 = I1*lofst*IN_ELEM_SZ + I2*16 +
                    I3*lofst*IN_ELEM_SZ + I4*IN_ELEM_SZ;

               (V_in1, Vin0) = data_ptr[A0].deinterleave();
```

```
            A1 = I3*coefw*COEF_ELEM_SZ + I4*COEF_ELEM_SZ;
            V_coef = coef_ptr[A1].onept();

            V_acc0 += V_in0 * V_coef;
            V_acc1 += V_in1 * V_coef;
          }
        }
        output_ptr[A2] =
       (Vacc1, Vacc0).interleave().round(16).saturate(16);
      }
   }
}
```

Figure 7: VCOP Kernel-C Program for 2D FIR

Analysis of the code shown in Figure 7 reveals a very C-like syntax that exposes all the capabilities of the underlying vector core hardware, without having to learn a lot of unique key words. The VCOP kernel-C syntax defines two data types __vector to convey 8-lanes x 40-bit = 320-bit vector register to hold the data and __agen to convey the 4-dimensional address generator resources available and unique to the vector core. This example leverages 4-levels of looping, which are supported by the vector core with zero-overhead, allowing counter maintenance and branch overhead to be completely hidden across all these 4-loop levels, in addition to having the ability to initialize multiple vector registers as well without consuming any additional cycles. The variable "VCOP_SIMD_WIDTH" designates the number of lanes of SIMD available on the core, which in the first generation of EVE is eight (8). The code reveals the flexibility of the load and store units to not only load the data, by accepting an address from a given address generator, but also to permute and re-arrange the data for free without bogging down the computational units. An example of this is the onept() load which can make a vector out of a scalar, whereas a de-interleave() load loads a register pair, while de-interleaving even data to one vector register and odd data to another register in a register pair. The VCOP vector core has two vector functional units, which can perform multiplication with optional rounding in addition to accumulation. The store operation can happen at any of the 4-loop levels and performs the re-packing of the data contained in the register pairs, in addition to supporting flexible rounding and saturation to any N-bit boundary. Performance of this code is given by (blkw x blkh x coefw x coefh)/16 + 50 cycles, independent of specific values of coefw and coefh, without making any specific assumptions on the filter taps. Figure 8, shows an example for 180 degree image rotation, leveraging the custom permutes on the load unit, along with the ability of address generators to apply either a positive/negative address increment, at every load/store.

```
void vcop_custom_load_test_cn
(
    __vptr_uint8       in,
    __vptr_uint8       out,
    int                offset
)
{
    __vector  data;

    for (int I2 = 0; I2 < (64/VCOP_SIMD_WIDTH); I2++)
    {
        __agen Addr_in, Addr_out;
        Addr_in   = (I2 *  VECTORSZ);
        Addr_out  = (I2 * -VECTORSZ);
        data      =  in[Addr_in].dist(7,6,5,4,3,2,1,0);
        (out + offset)[Addr_out]  =  data;
    }
}
```

Figure 8: VCOP Kernel-C Code for Image Rotation

7. CONCLUSIONS

This paper reviewed the TDA2x, a SoC for ADAS systems, which is built on a heterogeneous architecture with various programmable elements, with the TMS320C66x DSP and VisionAcceleratorPAC shouldering the challenging compute requirements found in vision analytics algorithms. EVE is a fully programmable vision analytics accelerator programmable from a templatized C++ language referred to as VCOP Kernel-C. Combination of the TMS320C66x DSP and a fully programmable vision accelerator in EVE allows for mapping challenging vision analytics algorithms on the TDA2x SoC while meeting challenging safety requirements.

12. REFERENCES

[1] TDA2x SoC for Advanced Driver Assistance Systems (ADAS) - TI.comwww.ti.com/pro-ap-tda2x-b-lp

[2] Arian Maghazeh, Unmesh D. Bordoloi Petru Eles Zebo Peng: General Purpose Computing on Low-Power Embedded GPUs: Has It Come of Age?, 13th International Conference on Embedded Computer Systems: Architectures, Modeling, and Simulation, Samos, Greece, July 15-18, 2013.

[3] Daniel Baumgartner, Peter Rossler and Wilfried Kubinger: Performance Benchmark of DSP and FPGA Implementations of Low-Level Vision Algorithms, IEEE Conference on Computer Vision and Pattern Recognition CVPR 2007.