Architectural Simulation for a Programmable DSP Chip Set

Jong Tae Lee, Jaemin Kim, and Jae Cheol Son

DSP Team, Micro Devices Business, Semiconductor Division, Samsung Electronics Co., Ltd.
San 14, Nongseo-Ri, Kihung-Yup, Yongin-Kun, Kyungki-Do 449-900, Korea
E-mail: {jtlee, jaemin, jcson}@jupiter.info.samsung.co.kr

Abstract— This paper presents an architectural simulator called VIDEOFLOW software tools for developing programmable DSP chip set for various video codec standards. This DSP chip set consists of the Image Compression Coprocessor (ICC) and Motion Estimation Coprocessor (MEC), which provide an easy solution for implementing the major digital video codec algorithms. The ICC/MEC simulation components are 100 percent bit accurate and closely approximate the timing of the actual chips. In addition, the simulation tool provides users with the ICC/MEC system simulation, debugging, and various performance monitors. This tool can also be used to define and modify the architectural specification for future product line of the ICC and MEC.

I. Introduction

Digital image processing is widely used in various applications such as TV transmission, video conferencing, computer communications, facsimile transmission, etc. Digital image compression has been one of the key areas of the digital image processing which is to reduce the storage and/or communication channel bandwidth needed to process such signals. Since these applications require different characteristics and environments, there are several standards for these applications. For example, ITU-T Recommendation H. 261 [1], Moving Pictures Experts Group (MPEG) [2], and Joint Photographic Experts Group (JPEG) [3] are used in video conferencing, motion picture, and still picture applications, respectively. In multimedia environment, it is more desirable for one system to support these standards. To support multiple standards, the system should have programmability and its architecture should be flexible. In addition, high performance is required because MPEG-1 video encoding requires approximately one giga operations per second (GOPS) without motion estimation [4]. The ICC [5] and MEC [6] chip set has enough capability to support these various digital video codec standards.

In the past, it has been common to first build a physical hardware prototype and then to integrate software and debug the system. With higher levels of integration, debugging the hardware prototype becomes much more difficult, and modifying the fabricated prototype requires additional cost and time consuming procedure. Furthermore, it is necessary to verify the interactions among multiple application specific integrated circuits (ASICs) and standard components before fabricating the ASICs. Since the ICC and MEC chip set and its peripherals are too complex, it is necessary to verify the functionality of the full system through simulation.

In this paper, we present an architectural simulator, VIDEOFLOW software tools. These tools provide a user-friendly interface to graphically write, assemble, link, simulate, and debug video compression algorithms such as JPEG, MPEG-1, and H.261. The simulator is for popular UNIX workstations running under X-Windows environment. It is implemented with C-language and Motif Graphical User Interface (GUI) tool. To simulate the parallel processing architecture, it employs event-driven simulation technology.

This paper is organized as follows; The ICC and MEC chip set is introduced in Section II. The VIDEOFLOW software tools are shown in Section III. The Simulator for the ICC/MEC and some utilities are discussed in Section IV and Section V, respectively, which is followed by Conclusions.

II. ICC and MEC Chip Set

The ICC chip, using its unique vector dataflow architecture, consists of eight asynchronous parallel processors on a shared 96-bit global bus. The maximum data throughput of the 96-bit global bus is 4.8 giga bits per second assuming the 50 MHz operation. The eight parallel processors operate under the supervision of an on-chip scalar processor called the Dataflow Control Unit (DCU). The DCU dispatches instructions stored in 128-word instruction memory, checking availability of input data and mapping instruction to a specific processor. Another unique feature of the ICC is that it employs vector dataflow programming using high-level instructions (e.g., forward discrete cosine transformation, inverse quantization, and forward run-length coding) while the the commonly available solutions are based on micro-codes. When hosted by popular RISC or DSP processors, the ICC and the host combination forms a complete highly programmable

computing solution for encoding and decoding still and motion video images taking advantage of its 1 GOPS performance.

The MEC is designed to be used in conjunction with the ICC in video compression applications, implementing standards such as MPEG-1 and H.261. The MEC is a very high performance parallel processor specifically optimized for real-time execution of motion estimation block matching algorithms. The MEC includes the Block Matching Processor of 7.2 GOPS which can compare three non-interpolated 16 by 16 blocks in 360 nsec or three 8 by 8 blocks in 100 nsec. The ICC treats the MEC as an external processing unit and uses vector instructions to send a reference pixel block to the MEC and receive the prediction result from the MEC. The MEC's on-chip RISC CPU responds to prediction requests from the ICC by executing the motion estimation algorithm written by users in "C". The MEC allows the use of virtually any type of the "mean of absolute difference" criterion-based full- or half-pel search algorithm, including exhaustive, two-level hierarchical, and some heuristic search algorithms. Multiple MEC chips can be connected to a single ICC for increasing motion estimation performance or search area.

The ICC/MEC chip set supports JPEG encoding or decoding of full resolution ITU-T-601 (720 by 480) images, simultaneous H.261 video encoding and decoding of CIF (352 by 288) images, or full MPEG-1 encoding or decoding of SIF (352 by 240) images, all at 30 frames per second. Using the 8.2 GOPS performance of the ICC/MEC chip set, a multi-point video conferencing which is capable of decoding seven incoming QCIF resolution streams in parallel with encoding of one QCIF stream is available.

III. The Videoflow Software Tools

The VIDEOFLOW software tools are to fully simulate the operations and timing of the ICC/MEC chip set. More specifically, users can quickly architect, program, simulate, and verify video compression solution based on the ICC and MEC chips prior to building any hardware. These tools are for UNIX workstation running under X-Windows. The bottom line is the followings: Users get to market quickly with low cost risk. Users are free to add their own special scheme for differenciating themselves from competitors.

Figure 1 shows the architectural block diagram of the VIDEOFLOW software tools. As we can see from Figure 1, the software tools consist of the following three major components:

- programming tools which let users easily create video compression programs for the ICC and MEC.
- simulation tools which let users easily create a simulation environment for an ICC/MEC-based compression system and use their programs to process bit streams.
- image analysis tools which let users examine simulation results by displaying processed video sequences with graphical data overlays on their workstation monitor.

A. Flowgraph Editor/Assembler/Linker

The Flowgraph Editor is a program created to enable the user to graphically write ICC program source code. Figure 2 shows the VIDEOFLOW Flowgraph Editor Window which consists of editor window, tool box window, internal instruction box window, and external instruction box window. The programs can be written by placing ICC instruction boxes on the screen and connecting the instruction boxes with arcs (directional lines indicating data flow). The high level ICC instructions enable the user to write complex algorithms with just a few instructions. For example, the JPEG decoding algorithm can be written using just five instructions as we can see from Figure 2. The ICC's flowgraph-based programming makes it easy for users to write programs that execute instructions in parallel. The user simply specifies how data flows between instructions and the ICC's dataflow controller does the rest. The controller automatically executes any instruction ready to run based on operand availability. The output of the Flowgraph Editor is a flowgraph file ready to be assembled.

The ICC Flowgraph Assembler is a program which assembles the flowgraphs created by the Flowgraph Editor. The output of the assembler is an object file ready to be linked. The ICC Flowgraph Linker is used to link object files assembled by the ICC Flowgraph Assembler. One or more object files may be linked to form an ICC program file which is executable on the Simulator.

B. MEC-CPU Compiler and Assembler

The MEC-CPU compiler and assembler produce MEC-CPU run time code from MEC-CPU program source code. The RISC CPU within the MEC implements the motion estimation algorithm. The user defines the motion estimation algorithm by writing an MEC CPU program. The output of the MEC-CPU compiler and MEC-CPU assembler is an MEC program file which can be run on the Simulator. Note that the MEC is a coprocessor to the ICC so an ICC program file must also be loaded for the Simulator to run an MEC program.

C. Host Interaction Language (HIL)

The HIL is a C-like interpreted language used to simulate the host processor. Both the ICC and MEC are slave processors and must be initialized by a host. The initialization includes resetting the coprocessors, loading registers and programs, and starting the coprocessors. The HIL program can also be used to transfer video data to the Simulator from the hard disk and vice-versa.

D. Simulator

The Simulator is a program used to simulate the ICC and MEC coprocessors. The simulator is designed to be accurate in timing as well as the execution of ICC and MEC functions. The Simulation Manger is the graphical user interface (GUI) to the Simulator and provides several tools that help the user configure the Simulator.

E. Utilities

The utilities are programs for viewing video images, disassembling ICC programs, showing ICC loading statistics, showing ICC instruction execution times, and converting zero-run length coded bit steams to MPEG-1, JPEG, or H.261 bit steams and vice-versa.

IV. Simulator

At the center of the VIDEOFLOW software tools is the Simulator and its GUI manager, the Simulation Manger. These tools provide functional and timing simulation of the ICC, MEC, video interface, and host interface.

The Simulator uses an event-driven technique to run the simulation. Each significant state change is one that affects the functional events or timing statistics that are required to be modeled. The simulator processes events in a time-ordered queue. It reads an event and passes it to the computer software unit designated to process it. The designated computer software unit performs both timing and functional processing and generates response events that are placed on the event queue for later processing.

The Simulator can be run from the UNIX command line or through the Simulation Manger in menu-driven windows environment.

A. Capability

The simulator models the ICC and MEC on both functional and timing levels. The Simulator allows users to run ICC and MEC programs jut as they would run on the actual IC chips. The Simulation Manger provides a GUI to the Simulator. It allows users to configure, run, debug, and examine simulations and their results. The Simulation Manger makes configuring a simulation easy by providing a menu-driven window interface.

Hardware configuration of the ICC and MEC, ICC register initialization, video memory configuration, and video timing generator (VTG) configuration can all be done by the Simulation Manager. Debugging features in the Simulation Manager allows users to locate errors in a program and examine data and status at any point in the ICC and MEC program.

Statistical outputs can be obtained at any point in a simulation. These outputs include timing information on the functional unit and instruction levels, and parallel execution count, ICC token memory usage, functional unit utilization, and parallel unit speed up. These statistics are useful in optimizing ICC data flow programs.

B. Operational Overview

The simulation process consists of the following steps:

- Launch the Simulation Manger from the Flowgraph Editor.
- Configure the Simulator.
- Configure the hardware environment.
- Configure the ICC and MEC internal registers.
- Execute the simulation: start, break, debug, continue, and stop as required.
- Start the dynamic bar graph statistical displays.
- Set up break-points as required for debugging.
- Examine statistics outputs and issue interactive HIL commands during the simulation.

B.1 Simulator Configuration

The Simulator must be configured in order to run properly. Options such as register database, ICC program, MEC program, and HIL program file names must be set.

B.2 Hardware Configuration

All hardware configuration parameters are set in the hardware configuration dialog box consisting of the following four parts each devoted to a specific configuration: video memory associated with the ICC and MEC, clock speeds, video timing generator, and auxiliary processors.

B.3 Register Configuration

ICC register initialization can be done by an HIL program or from a database file. The Simulator initializes all the registers in a register database file with the specified values. A register database file can define values for all registers except active registers.

B.4 Simulation Statistics

The Simulator produces a variety of statistics as a side effect of its operations, such as the simulation time, clock speeds for each asynchronous clock, total instruction processing time, parallel unit speedup, instruction statistics, functional unit statistics, data memory statistics, global bus statistics, and MEC statistics.

The Simulator accumulates a number of statistics pertaining to the ICC. These measurements are displayed in text format and in displays the change during the course of the simulation. Several graphic statistics displays are

also available within the Simulation Manager as like in Figure 3. They are provided in the form of histograms showing usage of the functional units, token memory, and the rate buffer FIFOs. The displays are updated with current information approximately every two seconds.

It is noted that these statistics result can be used to optimize the ICC internal components for some specific application. For instance, we can easily figure out from Figure 3 that the 75 percent of the token memory can be cut down for this specific application.

C. Simulation Debugging Facility

The debug facility of the Simulation Manger allows users to set breakpoints in the simulation path and examine the state of ICC instruction, ICC tokens, and the MEC CPU.

D. Run Simulation

Simulator runs through 5 components: simulation initialization, next-event determination, process simulation, statistics update, and statistics storage.

To initialize simulation, hardware configuration and timing data files are required. Once simulation is initiated, next-event determination unit fetches the next executable event from the event queue and triggers process simulation. Process simulation unit receives one simulation event and data. While processing one simulation event, process simulation unit generates response events and sent them to event queue. At the same time, process simulation unit calculates statistics which will be sent to statistics update unit and consequently to statistics storage.

E. Processor Simulation

The processor simulation consists of four major entities: process DCU events entity, process functional events entity, process host events and commands, and process auxililary unit events entity.

For each entity, an object, a structured data which contains processing timing characteristics, is defined to encapsulate all components necessary to describe it. This object is passed to software modules via events. In this way each module knows on which entity it is performing its operations.

V. Utilities

The Simulator also generates information to provide a full instruction trace. The program, called unit_scroll, relies on data gathered during a simulation run. It shows the instruction execution sequences on the ICC functional unit and auxiliary processors. The execution of an instruction is shown by time bar whose length represents the duration of processing as like in Figure 3. The time bar is labeled with the instruction's name and address in instruction memory. The vertical axis of the display is labeled with the processing units. Instructions that run in the same processing unit are horizontally aligned. A scroll bar lets you scroll horizontally through time to see the parallel execution of instructions on the various functional units. This feature combined with the statistics outputs can be used to optimize and modify the architecture for future product line of the ICC and MEC.

Image analysis tools let users visually and quantitatively evaluate the effectiveness of algorithms programmed on the simulated ICC/MEC chip set and embedded host processor. The image analysis tools consist of a video display utility combined with a novel with up to four different types of color-coded information like as in Figure 4. This information is automatically produced by MPEG or H.261 simulations and includes macroblock coding types, macroblock quantizers, and forward and backward motion vectors. The graphical overlays let users visually correlate this infomation with image pixels, letting users quickly evaluate their own compression algorithms in area such as mode selection, adaptive quantization, and motion estimation.

VI. Conclusions

This paper presented the VIDEOFLOW software tools for the simulation of the ICC and MEC chip sets. An event driven technique is used to run simulation. The Simulator allows the user to pause the simulation, issue debug commands, examine ICC internal data structure, and display simulation statistics. Using these tools, we can verify the the functionality of the system as well as a specific video codec algorithm. Since the simulator gives the information about the usage of each functional units (internal coprocessor), we can obtain information about how to optimize each functional unit for specific video codec standards.

References

[1] Ming Liou, "Overview of the $p \times 64$ video coding standard," *Communication of the ACM*, pp. 59-63, April 1993.

[2] Didiler Le Gall, "MPEG: A video compression standard for multimedia applications," *Communication of the ACM*, pp. 46-58, April 1993.

[3] Gregory K. Wallacc, "The JPEG still picture compression standard," *Communication of the ACM*, pp. 30-44, April 1993.

[4] Bang W. Lee et. al, "Data flow processor for multi-standard codec," *Proc. of ISSCC*, 1994.

[5] Samsung Electronics Co. Ltd., *SMP9601 Image Compression Coprocessor Data Sheet*, Rev. 1.2, March 1995.

[6] Samsung Electronics Co. Ltd., *SMP9602 Motion Estimation Coprocessor Data Sheet*, Rev. 1.1, March 1995.

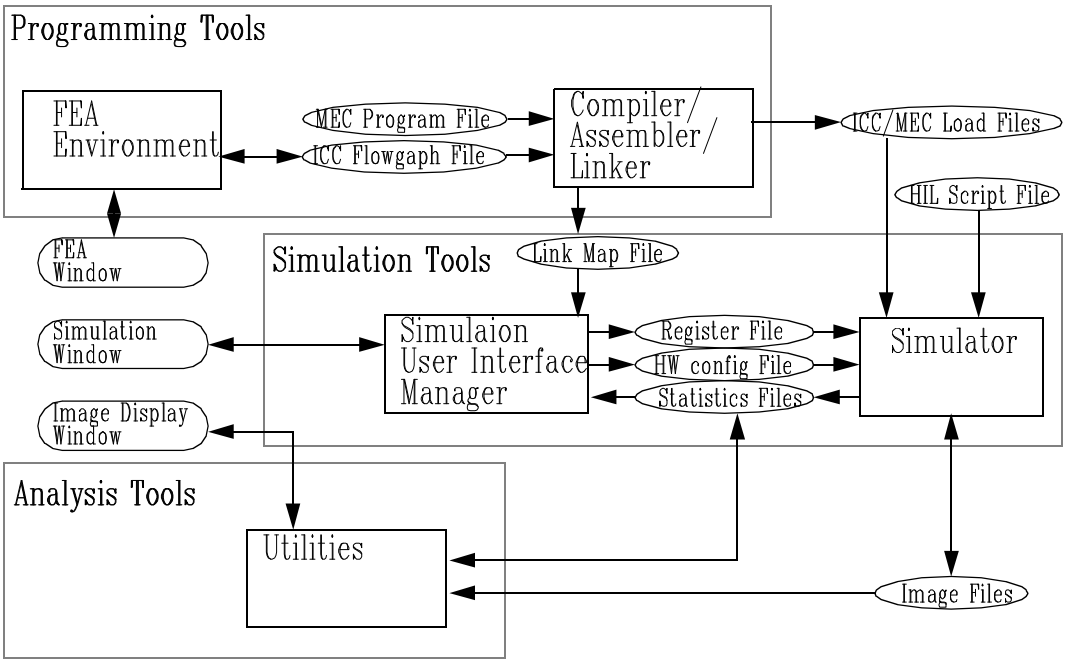

Fig. 1. Architectural Block Diagram of the VIDEOFLOW Software Tools

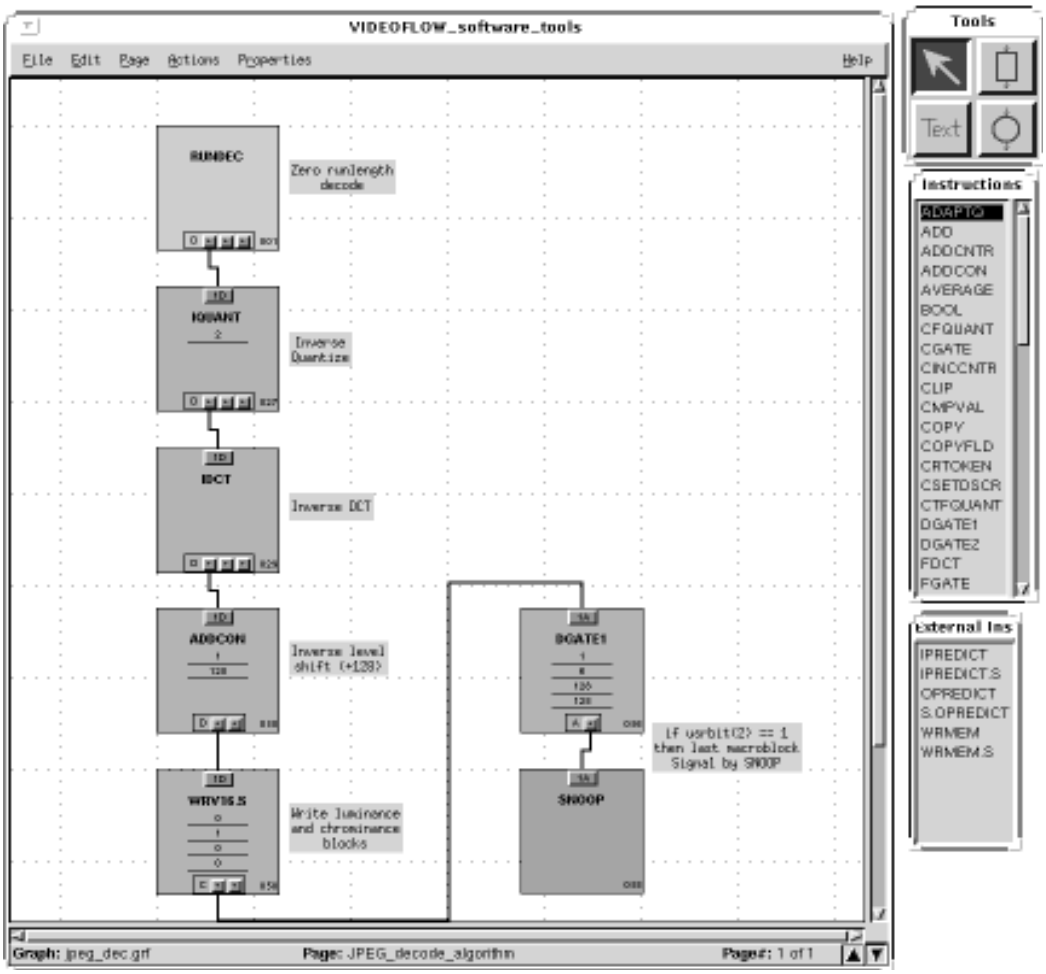

Fig. 2. The VIDEOFLOW Flowgraph Editor Window

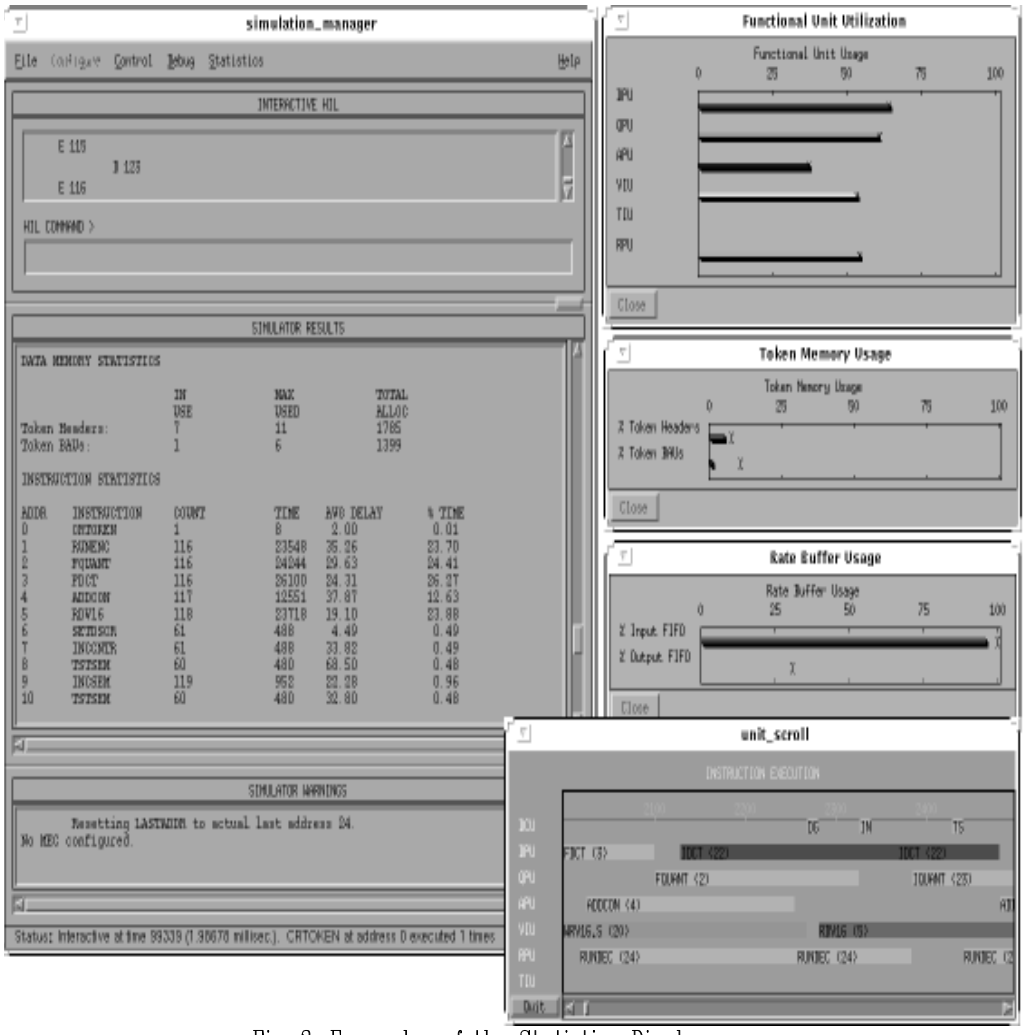

Fig. 3. Examples of the Statistics Displays

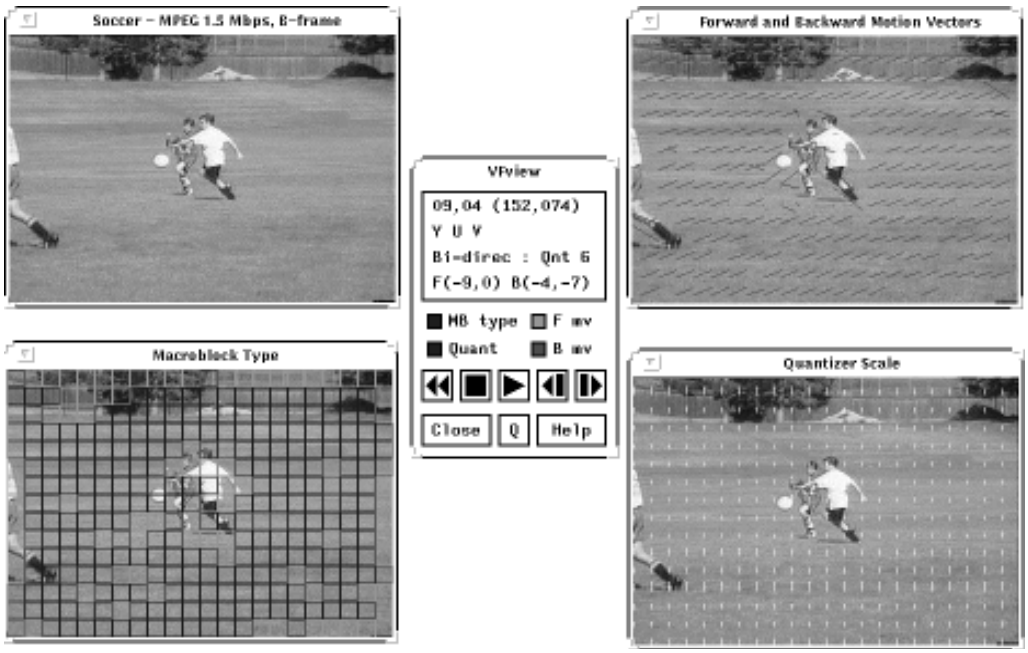

Fig. 4. Video Display Utility of the VIDEOFLOW Software Tools