Proceedings of the 5th WSEAS International Conference on Applied Computer Science, Hangzhou, China, April 16-18, 2006 (pp856-861)

Image Compression Preprocessor Design with task partitioning using coarse grained approach

MUHAMMAD KAMRAN, SHI FENG
Department of Computer Science and Engineering
Beijing Institute of Technology,
Beijing-100081
CHINA

Abstract: - For optimization of digital circuit, designers encounter number of problems regarding signal and time complexity which are required to be reduced or eliminated. During our research of designing image compression preprocessor, division of main task into comparatively smaller number of subtasks is carried out for the simplification of operation especially with respect to time. This paper presents the methodology to visualize efficient image compression preprocessor design on the basis of task partitioning considering coarse grain architecture. For the boosting of theoretical approach, recursive data extraction from image, purpose of z-transform utilization, signal flow graphs and time complexity for task completion is also incorporated in this paper. Moreover, practical solutions acquired by simulating tools are also assimilated in this paper with appropriate substantiation.

Key-Words: - task partitioning; z-transform; SFG; CWCR; coarse grain; simulation

1 Introduction

Presently, ASIC design is the area of research with lot of attention and consideration. The embedded systems are modified with new approaches and techniques to overall reduce the complexity cost and the size of resulting circuit. ASIC has crossed 10 million mark as per device count is concerned. Improvement in ASIC design is carried on using different mathematical methods of Petri nets, SFG and Hierarchical Concurrent Flow Graph (HCFG) [1]. The idea of using partitioning of system to make subsystems and utilizing Mason formula for the simplification of signal flow graph is grabbed from [7]. This paper describes that independent gains for all subtasks are obtained by taking the probability of task finishing time into consideration. After these calculations all sub modules are joined hierarchically to get over all system gain. After wards graph transmittance and expected task finish time is calculated. If components are large to be integrated back, may cause some error in timing and data flow signals. Concurrent flow model is a set of independent, encapsulated, concurrently operating model components where each model has its own thread of control and components [8]. The whole scenario is operated with minimum time clash between data and control signals to obtain most optimized solution. This approach looks effective to reduce the expected task finish time E [T_p]. The number and size of task in which problem is decomposed determines the granularity of the decomposition. Decomposition into large number of small tasks is called fine grained and decomposition into small number of large tasks is called coarse grained. If all tasks are performing their operations at the same time, system will attain maximum degree of concurrency. But maximum degree of concurrency is always less than the total number of tasks due to the dependency problem [2]. In the meanwhile, with all merits of parallel computation, there are few limitations as well. For example circuit complexity and area optimization is seriously effected. The remedy is to use pipelining structure for data transmission and extraction from source node and on wards. The constraint with pipeline structure design is the decay of efficiency of system with the increase in number of stages beyond a specified value (n) [3].

An effective approach of ASIC design is opted when total design is divided into small tasks and data extracted recursively and modules are operated with appropriate pipeline stages. SFG is evolved and graph transmittance and expected task finish time is calculated followed by simulation results. Design Rule Check (DRC) error should be zero after successful simulation to verify the proper placement and routing of design ready to mount on FPGA.

2 Hardware model

Hardware model block diagram of proposed concurrent preprocessor narrating different data and control signals is given in Fig. 1.

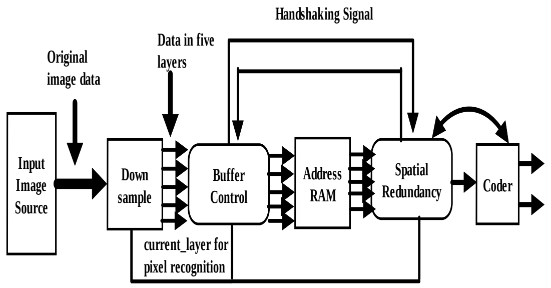
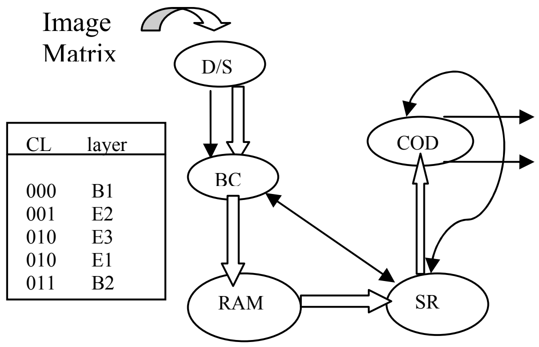


Fig. 1 Preprocessor with data and control signals

The image or picture is obtained from some input device and is applied to the preprocessor. This preprocessor performs the operation of spatial redundancy and divides image into number of layers. Afterwards each image pixel is stored in a given prescribed location with particular address. The RAM appended in the design is made very efficient to perform concurrent READ/WRITE operation. The size of RAM is to be made quite large as image pixels data is also very huge to be dealt with. In [4], it is suggested that image data has large number of frames and each frame is comprised of layers and each layer has number of pixels. As per operation of down sample module, the image is divided into five layers; that is why algorithm is also termed as layered algorithm. The dependency data flow graph (DDFG) of our design is given in Fig. 2. This figure not only describes the flow of data from one module task to the other but also gives the detail of control signals giving the detail of dependency. As it is apparent from Fig.2 that Buffer control (BC) module has dependency on spatial redundancy (SR) module and SR has dependency on coder. Dependency calculation evaluated from task decomposition choice plays an important role in the selection of a good mapping for a parallel algorithm. A good mapping should seek to maximize the use of concurrency by mapping independent tasks onto different processes [2]. The objective of proposed design prior to partitioning is described as follows;

The input data is to be compressed and to be obtained concurrently on the computer monitor screen to save the memory and to with stand bandwidth variation in the system. The proposed model of preprocessor will provide two independent routs for pixels and connected to two coders operating on DCT algorithm. Following are the task partitioning steps for successful coarse grained partitioning so called because large system is to be divided into comparatively small number of tasks;

- The image pixel data is the bmp file contained in a matrix and arrives down sample module in a certain sequence.
- All pixels belong to one of the defined layers; initially 3 enhanced and 1 base layer.
- Base layer B2 is extracted from E1, and resides in the pixel recognition table with the help of control signal, current layer (CL).
- After getting the generalized idea of first task module (D/S), there is a serious requirement of such a module which generates required address of the location to place the pixels belonging to their corresponding layer number and type. Fig.2 also gives the CL number which also acts as the recognition number of layers in the design.



CL	layer
000	B1
001	E2
010	E3
010	E1
011	B2

Fig. 2 DDFG with signal dependencies

- Buffer control (BC) unit helps to calculate the corresponding address in PING PONG RAM to place the pixel in its appropriate position.
- Next is to place complete data frame (5 Layers) into the RAM and after wards extracted for motion estimation unit. In the meanwhile BC has handshaking with SR to get permission of next data transmission.
- (SR) plays a vital role to bifurcate the data into E's (Enhanced layers) and B's (Base layers), which are further transmitted to coder for image compression algorithm processing.

Decomposition of tasks is of different types like recursive, data, exploratory and speculative decomposition. For our design recursive partitioning is suitable as data is picked up from the input matrix randomly depending upon pixel correspondence with current layer value.

Data handling and correspondence is described on the bases of sorting algorithm. Fig.3 shows the recursive decomposition for sorting a sequence of 64 pixels in one image frame. It explains the complete process of partitioning and pixel placement in optimized RAM with two frames only.

It performs concurrent READ and WRITE data operation. Output is bifurcated and rebuilding of B2 pixels take place in CODER, the last module in our design. SR performs the pixel prediction process for MPEG compression and spatial redundancy for JPEG compression as mentioned in [5].

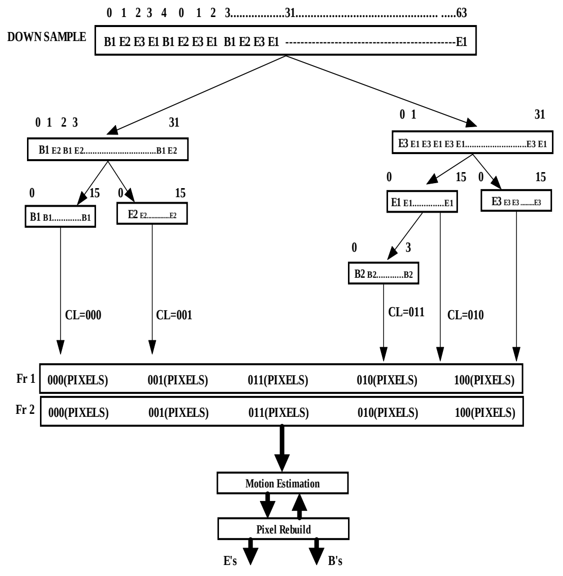


Fig. 3 DOWN SAMPLE process pixel sorting, DUAL frame RAM, Motion Estimation, Pixel Rebuild and 2 outputs (E's and B's)

Fig.3 suggests that all modules can work independently but some modules wait for the completion of task from previous module and then perform their own operation. It means some inevitable dependency of signal is also found in the design. For example, SR module can not be operated till rebuild data from coder module goes back into it for the extraction of new enhanced layers to be displayed on one of the out puts. Also SR module has dependency on BC module that new pixel data should not be forwarded to SR till previous calculation and estimation is finished. Indirectly, we can suggest that BC depends upon SR, and SR depends upon CODER. It means BC is indirectly depends upon CODER.

Another useful method to represent a system with constraints is to make Design structure matrix (DSM), which is considered as simplest representation of digital design showing the dependencies and direction of flow of one task to the others. Fig.4 gives DSM enabling us to describe our model while X shows the dependency of one task onto the other.

Activities	A	B	C	D	E
Down sample(A)	**A**				
Buffer control (B)	X	**B**			
RAM (C)		X	**C**		
SR (D)		X	X	**D**	
Coder (E)				X	**E**

Fig. 4 DSM of our preprocessor design

3 Circuit Complexity

There are many techniques available to construct the digital circuit. The criteria are to trade off cost and get circuit scalability with performance. The simplest and common to all is bus topology, which is utilized in given design. Fig.5 gives the general layout of the system based on bus topology.

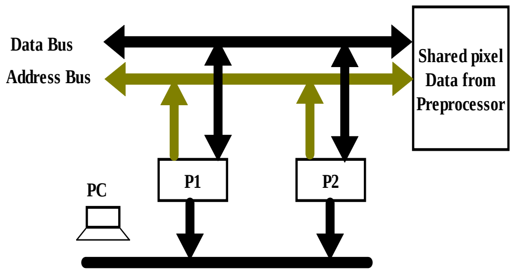


Fig. 5 Concurrent Compression through 2 processors on single PC (Bus Topology)

Fig.5 suggests that pixel data extraction from preprocessor and available on bidirectional data bus. Address line is unidirectional as address is generated in DCT processors as per our design and data is extracted according to the address from memory. This topology is also CWCR (Concurrent write concurrent read) based, i.e., two processors are sharing their data from one memory source in parallel [6]. The preprocessor represents the overall function of memory and data propagation from input image source. From two coders, data is obtained concurrently and are displayed on PC. Designers can have an option to use concurrent compression concept with single processor as well if its processing speed is reasonably high. The processors or coder designs are not in the scope of this research paper. The cost of the algorithm and operation is normally taken in terms of time and area occupied. It is proved fact that different design configurations attain different complexity. The time cost of communication depends on variety of features including programming model semantics, circuit topology, data handling and routing and associated software protocols.

Time taken to communicate between two nodes in a network is the sum of the time to prepare message and time taken by message to traverse the network to its destination. For our design we prefer store and forward routing whose communication cost is written as in (1);

$$t_{comm} = ts + (mt_w + t_h) * l \qquad (1)$$

$$ts = Startup\ time;$$

$$t_h = per - hop\ time;$$

$$t_w = word\ transfer\ time;$$

$$m, l \rightarrow message\ size, links$$

3.1 Z-transform application

For the calculation of time cost, main idea is extracted from HCFG as described in [1]. From dependency diagram, all nodes are connected in the parallel or in series combination. Fig. 6 is flow graph (FG) of task providing detail of expected task completion time and dependency flow. As task expected time can be described in terms of z-transform and rate of change of node transmittance, the option to represent transmittance using Mason's rule and task expected finish time as, $E[T_p]=z.\ d\Gamma/dz$ is utilized.

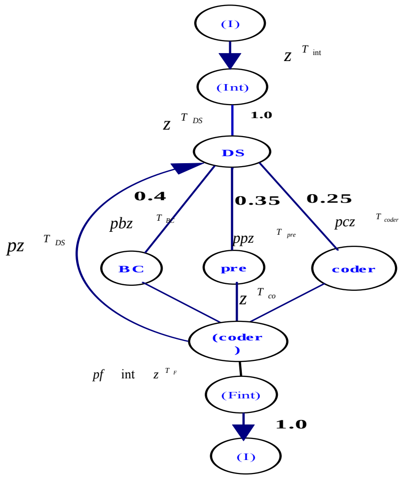


Fig. 6 Flow graph Model of Our Design

Fig.6 suggests that after down sampling the data, remaining 3 tasks are performing their operations concurrently. For this proposed design, $Z^T = t_{comm}$, data transmission time from one task to the other, while "p" is the probability of repeating the entire flow due to timing failure detected during layout verification. The transmittance of our design is given in (2) evaluated by Mason's formula, which further helps us to calculate expected task finish time.

$$\Gamma_{I,F} = \frac{P_{f\,\mathrm{int}} Z^{(T_{\mathrm{int}}+T_{DS}+T_{CO}+T_F)}(pcZ^{T_{CODER}} + pbZ^{T_{BC}} + ppZ^{T_{PRE}})}{1 - pZ^{(T_{DS}+T_{CO})}(PbZ^{T_{BC}} + PpZ^{T_{PRE}} + PcZ^{T_{CODER}})} \qquad (2)$$

Expected transmission time from node (I) to node (F) is given by formula in (3) by taking z value equals to 1;

$$E[T_p] = \frac{d\Gamma_{I,F}}{dz} \qquad (3)$$

Z-transform has one important application of representing delay in the signal depending upon negative power of z. If z^{-1}, means one time unit delay and so on. Moreover, it can be differentiated for the calculation of inverse z-transform, when X (z) contains multiple order poles. For data communication and speed calculation, it is found that speed of processor on CHIP is different from the speed of processor in computer's mother board. For data communication from DSP via PCI is obtained with data width of 32 bits and processing speed of 33 MHz. On our prepared chip, DSP will be available with more than one SD RAM as DSP cache is not enough to store all instructions executed with such a high processing speeds. HS is the handshaking signal available to coordinate between DSP and computer processor which will enable to accept next data of previous one is displayed. Block diagram description is given in Fig. 7, which depicts the rough idea of concurrent compression through processor with two images and two A/D converters available at a time on the input.

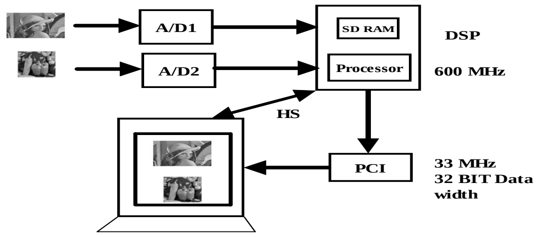


Fig. 7 Representation of concurrent Image applied and displayed on PC

For proposed design, 64 pixel data is sent for one frame at the rate of 20 ns/pixel, and on the receiving end 64 pixels are divided into 5 layers plus additional pixels during rebuilding process will cause 16 pixels in each layer and will deal with 2/5 X 80 with same time rate due to concurrent compression in this coarse grained design approach. After having rough evaluation of t_s, t_w and t_h is calculated respectively to estimate total communication task time for store and forward routing. The value of per hop time in parallel computers is smaller than the value of mt_w. If band width BW of channel is r words/sec, then each word takes time $t_w=1/r$ to traverse the link.

Ping Pong RAM pseudo code is already presented in [4]. This RAM worked perfectly with dual frames for concurrent READ/WRITE operation. The simulation of large size RAMS is possible in a reasonably small time, but its synthesis stuck and goes into infinite time loop and ultimately fails. The solution of these problems are while working on FPGA to use IP CORE launched by the company whose device is used or for ASIC external RAM is utilized in the design as depicted in Fig. 7.

Table 1 Path probability and task time

Probability	Path Time
P_{BC} = 0.4	T_{DS} = 5
P_{Coder} = 0.25	T_{BC} = 2
P_{Pre} = 0.35	T_{Pre} = 3
P_v = 0.10	T_F = 4
…………………..	T_{int} = 2
P_{fint} = 0.9	T_{CO} = 4

4 Experimental results

After investigating the problem of coarse grained partitioning, pixel extraction and access using recursive decomposition and transmittance of data using FG, it is concluded that time complexity of data transmission from one node to other in flow graph (FG) is low if concurrent module operation is implicated in design. In the signal flow graphs, all tasks are performed in a sequence depending upon the dependency, while in improved FG maximum tasks are to be considered to be finished in parallel and their results are incorporated after wards to get complete expected task finish time. Although inevitable dependencies are also present in CFG representation, but its task finishing time is still less than normal FG method. The probability of BC unit finishing time is assumed maximum, which will finish its task in minimum time, while coder module probability of finishing task is minimum will take maximum time to finish its job. This situation is well elaborated in Fig. 8 with BC, SR and coder probabilities are taken as 0.4, 0.35 and 0.25 respectively.

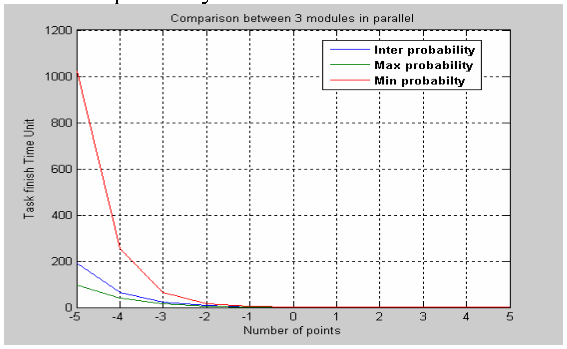


Fig. 8 Number of points and their z-transform representing time units for task completion

The graph transmittance of our preprocessor is given in (4), which helps to calculate time units for E_{Tp}. For these calculations, keeping our practical experience in view, probabilities of all tasks are assumed as per insight in designed CFG. All path times and probabilities are listed in Table 1. The results are presented in Fig.8; clearly give the time optimization on the bases of z-transform extracted from Fig.6. By substituting the values of probabilities and task time from table 1 into (2) and (3), graph transmittance and expected task finish time are obtained respectively.

$$\Gamma_{I,F} = \frac{0.9Z^{19}(0.25Z^4 + 0.35Z^3 + 0.4Z^2)}{1 - 0.1Z^9(0.25Z^4 + 0.35Z^3 + 0.4Z^2)} \tag{4}$$

Expected task finish time of pre processor after differentiating transmittance is, E $[T_p]$ = 22.25 units.

If comparison is made between proposed CFG flow graph with a graph in which all modules operate sequentially, of course path from input node to output node will be increased with their corresponding edge values and high E $[T_p]$ value is obtained.

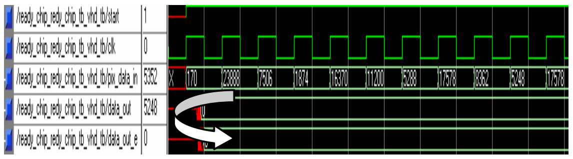
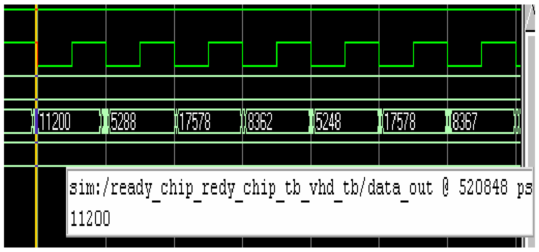

/ready_chip_redy_chip_tb_vhd_tb/start 1
/ready_chip_redy_chip_tb_vhd_tb/clk 0
/ready_chip_redy_chip_tb_vhd_tb/pix_data_in 5352
/ready_chip_redy_chip_tb_vhd_tb/data_out 5248
/ready_chip_redy_chip_tb_vhd_tb/data_out_e 0
X 170 23888 7506 1874 16370 11200 5288 17578 8362 5248 17578
U
U

Fig. 9 Input data in decimal while two output data in wait state till 500 ns (output is not visible yet)



```
sim:/ready_chip_redy_chip_tb_vhd_tb/data_out @ 520848 ps
11200
```

Fig. 10 Output Data appears after 520 ns

Fig.9 and Fig.10 provides us the post simulation results of proposed design after considering 16 bit assumed pixel data. MODEL SIM simulation tool is used for the result analysis. Fig. 9 is representing the input data applied for the verification of design and bold arrow indicates the space of representing out put data, not visible in figure till 500 ns. Fig. 10 is representing the output of design simulation with reasonably good and expected results. It is evolved from diagram that data pixels appearing on the output are about 90% same as those input pixels. Few unknown pixels are also added during image rebuilding process but they do not affect the quality of image. In the meanwhile, pixels which disappear from the input are compensated by new generated pixels during simulation.

5 Conclusion and Future Work

This paper gives guideline for the researchers in the area of algorithm design and verification on the bases of graph theory. Graph theory has the natural ability to deal with the adjacency of the nodes/tasks so that we evaluate the time complexity of the tasks and hence the complete digital system. The proposed pre processor will be appended with video coders and will get image input data concurrently. This circuit can be implemented on FPGA and can also be embedded for video compression ASIC. Its design and operation is verified using MODEL SIM simulating tool by the virtue of XILINX ISE with signal frequency 50 MHz. In future it is planned to calculate noise or error transmission with signals and its elimination. As in Fig.10, some anonymous pixels are contributing to the output, which are required to be estimated whether these pixels are part of data or add some noise in image transmission. Data noise detection and elimination techniques are available but new innovative mathematical approach of calculating condition number of pixel data, and its careful analysis will help to control error propagation in the system. MATLAB tool box can also be effective to recognize the image noise so that hardware designers can fix the problem with respect to their designs and verify the results.

References

[1] Vineet Sahula, C. P. Ravikumar and Nagchoudhuri, Improvement of ASIC Design Processes, *Proceedings of 15th International conference on VLSI design 2002*

[2] Ananth Grama, Anshul Gupta et.al, *Introduction to Parallel Computation, 2nd Edition,* China Machine Press, 2003.

[3] Kai Hawang, *Advanced computer Architecture: Parallelism, Scalability, Programmability,* McGraw Hill Inc., International Editions 1993.

[4] Muhammad Kamran, Shi Feng, Ji Weixing, Large Data Handling Techniques for compression Pre coder Chip using Scalable Algorithm, *Journal of Information and Communication Technology, Vol.1, No.1 (summer 2005) pp 47-54.*

[5] Muhammad Kamran, Shi Feng, Suhail Aftab, Data Acquisition and Handling for the Simulation of ASIC by utilizing SW/HW Co-Design Methodology, *IEEE ICET2005 proceedings, September 2005*

[6] Ralph Duncan, A Survey of parallel computer Architectures, *Computer Volume 23, Issue 2, Feb. 1990 Page(s):5 – 16.*

[7] Marwan M. Hassoun, Hierarchical Symbolic Analysis of Large-scale Systems Using A Mason's Signal Flow Graph Model, *Circuits and Systems, 1991., IEEE International Symposium on 11-14 June 1991 Page(s):802 - 805 vol.2*

[8] Douglas G. Fritz and Robert G. Sargent, An Overview of Hierarchical control flow Graph models, *IEEE Simulation Conference Proceedings, 1995. Winter 3-6 Dec. 1995 Page(s):1347 - 1355*