REALIZATION OF ARRAY ARCHITECTURES FOR VIDEO COMPRESSION ALGORITHMS

Yeu-Shen Jehng, Liang-Gee Chen, Tzi-Dar Chiueh, and Thou-Ho Chen

Room 416, Department of Electrical Engineering,
National Taiwan University, Taipei, Taiwan 10764, R. O. C.

Abstract

In this paper, a practical design technique is presented to realize the array architecture for hierarchical block matching algorithms. A mapping procedure has been applied to derive the array processor from the algorithm. The proposed systolic array is derived to reduce the I/O bandwidth and the hardware cost. This systolic array is configurated to be single-chip or cascaded architectures to match the real-time video applications.

I. Introduction

In motion estimation, the three-step hierarchical search algorithm [1] (3HSA) has fewer computational overheads than the full search algorithm [2-4] to find the motion vector. However, the computation of 3HSA includes some limitations which cause its hardware implementation lots of difficulties. These limitations are: i) the computation of next step should not start until the completion of the current step and ii) the sizes of searching window in each step are different, which will result in different data accessing modes. Therefore, it is attractive to develop an unified architecture to overcome the above limitations. In this paper, an efficient mapping procedure [5] has been applied to derive the systolic array architecture. The primary goal is to derive a common low-latency and low-I/O- bandwidth array architecture for each individual step. Consequently, this unified architecture can easily be extended to compute 3HSA either in single-chip or in cascaded fashions depending on the performance requirements of application systems.

II. The derivation of systolic architectures from 3HSA

For extracting the primitive operation, the single-stepped algorithm of N×N 3HSA is written as follows:

For i = 0, d, 2d
For j = 0, d, 2d

$$MAD(i,j) = \sum_{n=0}^{N-1} \sum_{k=m=0}^{N-1} | X(n,m) - Y(n+i,k+j) |;$$

$$V_S = (i,j) \mid \min MAD(i,j);$$

In the above algorithm, d is equal to $2^{(3-s)}$ in case the step s is executed where s may equal to 1,2, and 3. In this way, each step of 3HSA is unified. The dependence graph (DG) of this unified algorithm can be obtained by taking the single assignment code of this algorithm. For simplicity, we suppose the block size N to be 8 and the three-dimensional (3-D) DGs corresponding to each step of 3HSA can therefore be derived as shown in Figure 1-Figure 3. Note that the functions of processing nodes are also shown in those Figures. In these DGs, the referenced block Xs are transmitted through direction $[n,m,k]^T=[0,0,1]^T$ every $2^{(3-s)}$ units while the candidate block Ys are transmitted through direction $[0,1,-1]^T$. The

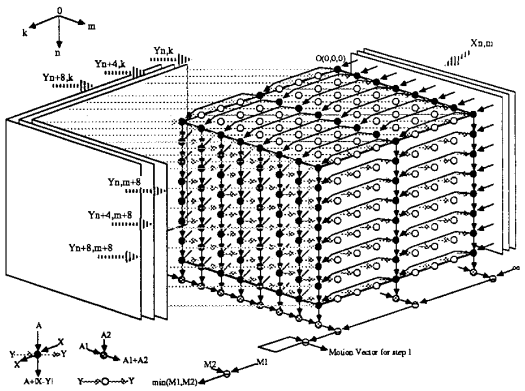


Figure 1. The dependance graph for step 1 of 3HSA.

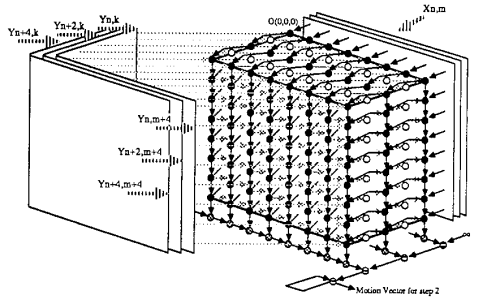


Figure 2. The dependance graph for step 2 of 3HSA.

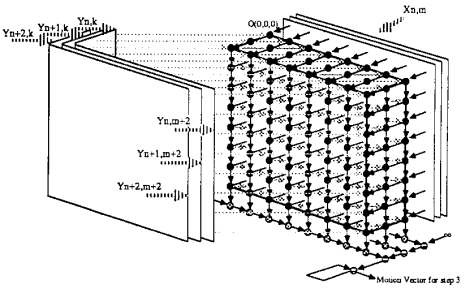


Figure 3. The dependance graph for step 3 of 3HSA.

0-7803-0593-0/92 $3.00 1992 IEEE

absolute-differenced data are accumulated through direction $[1,0,0]^T$ and are further summed up in plane $[N,m,k]^T$. At last, the 9 candidate blocks in the exact step of 3HSA are all matched against the referenced block so that the candidate block with minimum distortion is determined. Considering the data precedence and the design complexity of PEs, all DGs are projected in direction $[1,0,0]^T$ so that their corresponding 2-D signal flow graphs (SFGs) can be obtained. The 2-D SFG for step 1 of 3HSA is illustrated in Figure 4. In this SFG, the most critical limitation for VLSI implementations is in its high I/O bandwidth. Therefore, we further map those 2-D SFGs in direction $[m,k]^T=[0,1]^T$ in order to obtain the low-latency and the low-I/O-bandwidth 1-D systolic architectures. The individual architecture for each step of 3HSA and the corresponding computing sequences are shown in Figure 5-Figure 7. It is clearly that these designs provide the perfect unified architecture and adequate timing sequence. In those computing sequences, the marked Ys indicate that their absolute-differenced data are accumulated by the transmitted PEs when the exact time steps are reached; otherwise, the unmarked Ys indicate that they are just transmitted to their next PEs. From these obtained 1-D systolic architectures, obviously, they are similar to each other. This is the reason why two projections, in direction $[1,0,0]^T$ and in direction $[0,1]^T$ respectively, are the better choices. At last, by applying appropriate controls, the designed systolic architecture is shown in Figure 8(a) and the corresponding PE design is shown in Figure 8(b). One 11-bit adder and one absolutor are designed in the PE. Note that there is a cyclic register set which is used for supporting the data reuses of Xs so as to reduce the I/O bandwidth. The Xtag and the Ytag are used for controlling the PE to either function as a difference accumulator or a shifter. Because the architecture is unified in each individual step, it can be easily organized either in a single-chip structure (Figure 9(a)) to sequentially execute the three steps of 3HSA or in a cascaded structure (Figure 9(b)) to simultaneously execute those steps of different block matchings to meet higher performance requirements as shown in Figure 9(c).

III. Performance analyses

From the hardware design point of view, the performance of a video compression algorithm is measured at least by the following criteria: computation complexity, regularity of data flow, and the cost of real-time implementation. The proposed unified architecture can provide excellent simplicity and regularity. Considering real-time implementation, the performance can be significantly improved. In the single-chip system, the latency required to execute one step s of N×N 3HSA is $[3(N+2d)N+N]T_c$ where $d=2^{(3-s)}$, s=1,2,3, and T_c is the clock cycle time. Thus, the total latency is $(9N^2+45N)T_c$. By applying a video format : X×Y frame size, N×N block size, and R Hz frame rate to this single-chip system, we obtain a constraint as follows:

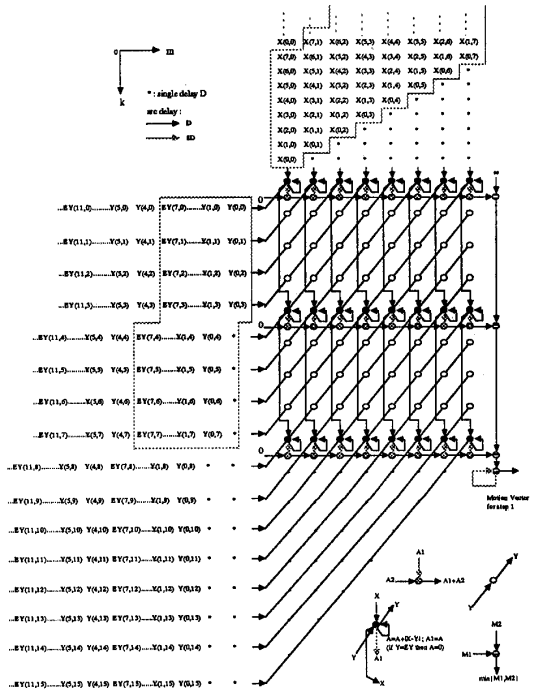
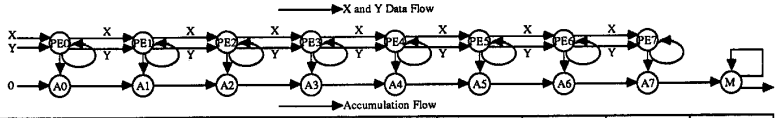


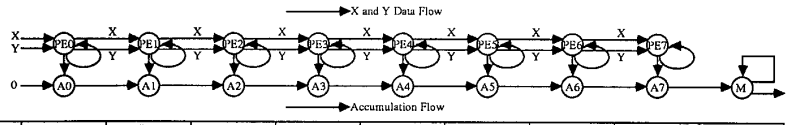
Figure 4. The 2-D signal flow graph for step1 of 3HSA after projection in direction $[1,0,0]^T$



time step n	PE0	PE1	PE2	PE3	PE4	PE5	PE6	PE7	M
	n ~ n+7	n+1~n+8	n+2~n+9	n+3~n+10	n+4~n+11	n+5~n+12	n+6~n+13	n+7~n+14	n
0	Y(0,0)-Y(7,0)	Y(0,0)-Y(7,0)	Y(0,0)-Y(7,0)	Y(0,0)-Y(7,0)	Y(0,0)-Y(7,0)	Y(0,0)-Y(7,0)	Y(0,0)-Y(7,0)	Y(0,0)-Y(7,0)	
8	Y(0,1)-Y(7,1)	Y(0,1)-Y(7,1)	Y(0,1)-Y(7,1)	Y(0,1)-Y(7,1)	Y(0,1)-Y(7,1)	Y(0,1)-Y(7,1)	Y(0,1)-Y(7,1)	Y(0,1)-Y(7,1)	
16	Y(0,2)-Y(7,2)	Y(0,2)-Y(7,2)	Y(0,2)-Y(7,2)	Y(0,2)-Y(7,2)	Y(0,2)-Y(7,2)	Y(0,2)-Y(7,2)	Y(0,2)-Y(7,2)	Y(0,2)-Y(7,2)	
24	Y(0,3)-Y(7,3)	Y(0,3)-Y(7,3)	Y(0,3)-Y(7,3)	Y(0,3)-Y(7,3)	Y(0,3)-Y(7,3)	Y(0,3)-Y(7,3)	Y(0,3)-Y(7,3)	Y(0,3)-Y(7,3)	
32	Y(0,4)-Y(7,4)	Y(0,4)-Y(7,4)	Y(0,4)-Y(7,4)	Y(0,4)-Y(7,4)	Y(0,4)-Y(7,4)	Y(0,4)-Y(7,4)	Y(0,4)-Y(7,4)	Y(0,4)-Y(7,4)	
40	Y(0,5)-Y(7,5)	Y(0,5)-Y(7,5)	Y(0,5)-Y(7,5)	Y(0,5)-Y(7,5)	Y(0,5)-Y(7,5)	Y(0,5)-Y(7,5)	Y(0,5)-Y(7,5)	Y(0,5)-Y(7,5)	
48	Y(0,6)-Y(7,6)	Y(0,6)-Y(7,6)	Y(0,6)-Y(7,6)	Y(0,6)-Y(7,6)	Y(0,6)-Y(7,6)	Y(0,6)-Y(7,6)	Y(0,6)-Y(7,6)	Y(0,6)-Y(7,6)	
56	Y(0,7)-Y(7,7)	Y(0,7)-Y(7,7)	Y(0,7)-Y(7,7)	Y(0,7)-Y(7,7)	Y(0,7)-Y(7,7)	Y(0,7)-Y(7,7)	Y(0,7)-Y(7,7)	Y(0,7)-Y(7,7)	
64	Y(0,8)-Y(7,8)	Y(0,8)-Y(7,8)	Y(0,8)-Y(7,8)	Y(0,8)-Y(7,8)	Y(0,8)-Y(7,8)	Y(0,8)-Y(7,8)	Y(0,8)-Y(7,8)	Y(0,8)-Y(7,8)	
72	Y(0,9)-Y(7,9)	Y(0,9)-Y(7,9)	Y(0,9)-Y(7,9)	Y(0,9)-Y(7,9)	Y(0,9)-Y(7,9)	Y(0,9)-Y(7,9)	Y(0,9)-Y(7,9)	Y(0,9)-Y(7,9)	
80	Y(0,10)-Y(7,10)	Y(0,10)-Y(7,10)	Y(0,10)-Y(7,10)	Y(0,10)-Y(7,10)	Y(0,10)-Y(7,10)	Y(0,10)-Y(7,10)	Y(0,10)-Y(7,10)	Y(0,10)-Y(7,10)	
88	Y(0,11)-Y(7,11)	Y(0,11)-Y(7,11)	Y(0,11)-Y(7,11)	Y(0,11)-Y(7,11)	Y(0,11)-Y(7,11)	Y(0,11)-Y(7,11)	Y(0,11)-Y(7,11)	Y(0,11)-Y(7,11)	
96	Y(0,12)-Y(7,12)	Y(0,12)-Y(7,12)	Y(0,12)-Y(7,12)	Y(0,12)-Y(7,12)	Y(0,12)-Y(7,12)	Y(0,12)-Y(7,12)	Y(0,12)-Y(7,12)	Y(0,12)-Y(7,12)	
104	Y(0,13)-Y(7,13)	Y(0,13)-Y(7,13)	Y(0,13)-Y(7,13)	Y(0,13)-Y(7,13)	Y(0,13)-Y(7,13)	Y(0,13)-Y(7,13)	Y(0,13)-Y(7,13)	Y(0,13)-Y(7,13)	
112	Y(0,14)-Y(7,14)	Y(0,14)-Y(7,14)	Y(0,14)-Y(7,14)	Y(0,14)-Y(7,14)	Y(0,14)-Y(7,14)	Y(0,14)-Y(7,14)	Y(0,14)-Y(7,14)	Y(0,14)-Y(7,14)	
120	Y(0,15)-Y(7,15)	Y(0,15)-Y(7,15)	Y(0,15)-Y(7,15)	Y(0,15)-Y(7,15)	Y(0,15)-Y(7,15)	Y(0,15)-Y(7,15)	Y(0,15)-Y(7,15)	Y(0,15)-Y(7,15)	
128	Y(4,0)-Y(11,0)	Y(4,0)-Y(11,0)	Y(4,0)-Y(11,0)	Y(4,0)-Y(11,0)	Y(4,0)-Y(11,0)	Y(4,0)-Y(11,0)	Y(4,0)-Y(11,0)	Y(4,0)-Y(11,0)	
136	Y(4,1)-Y(11,1)	Y(4,1)-Y(11,1)	Y(4,1)-Y(11,1)	Y(4,1)-Y(11,1)	Y(4,1)-Y(11,1)	Y(4,1)-Y(11,1)	Y(4,1)-Y(11,1)	Y(4,1)-Y(11,1)	

marked : for Accumulations unmarked : for Y Shifting

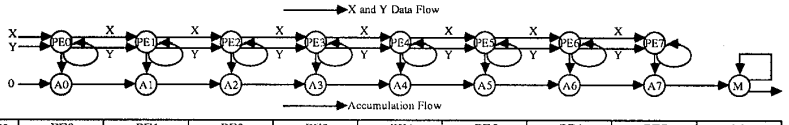
Figure 5. The 1-D systolic architecture and the corresponding computing sequence for step 1 of 3HSA



time step n	PE0	PE1	PE2	PE3	PE4	PE5	PE6	PE7	M
	n ~ n+7	n+1~n+8	n+2~n+9	n+3~n+10	n+4~n+11	n+5~n+12	n+6~n+13	n+7~n+14	n
0	Y(0,0)-Y(7,0)	Y(0,0)-Y(7,0)	Y(0,0)-Y(7,0)	Y(0,0)-Y(7,0)	Y(0,0)-Y(7,0)	Y(0,0)-Y(7,0)	Y(0,0)-Y(7,0)	Y(0,0)-Y(7,0)	
8	Y(0,1)-Y(7,1)	Y(0,1)-Y(7,1)	Y(0,1)-Y(7,1)	Y(0,1)-Y(7,1)	Y(0,1)-Y(7,1)	Y(0,1)-Y(7,1)	Y(0,1)-Y(7,1)	Y(0,1)-Y(7,1)	
16	Y(0,2)-Y(7,2)	Y(0,2)-Y(7,2)	Y(0,2)-Y(7,2)	Y(0,2)-Y(7,2)	Y(0,2)-Y(7,2)	Y(0,2)-Y(7,2)	Y(0,2)-Y(7,2)	Y(0,2)-Y(7,2)	
24	Y(0,3)-Y(7,3)	Y(0,3)-Y(7,3)	Y(0,3)-Y(7,3)	Y(0,3)-Y(7,3)	Y(0,3)-Y(7,3)	Y(0,3)-Y(7,3)	Y(0,3)-Y(7,3)	Y(0,3)-Y(7,3)	
32	Y(0,4)-Y(7,4)	Y(0,4)-Y(7,4)	Y(0,4)-Y(7,4)	Y(0,4)-Y(7,4)	Y(0,4)-Y(7,4)	Y(0,4)-Y(7,4)	Y(0,4)-Y(7,4)	Y(0,4)-Y(7,4)	
40	Y(0,5)-Y(7,5)	Y(0,5)-Y(7,5)	Y(0,5)-Y(7,5)	Y(0,5)-Y(7,5)	Y(0,5)-Y(7,5)	Y(0,5)-Y(7,5)	Y(0,5)-Y(7,5)	Y(0,5)-Y(7,5)	
48	Y(0,6)-Y(7,6)	Y(0,6)-Y(7,6)	Y(0,6)-Y(7,6)	Y(0,6)-Y(7,6)	Y(0,6)-Y(7,6)	Y(0,6)-Y(7,6)	Y(0,6)-Y(7,6)	Y(0,6)-Y(7,6)	
56	Y(0,7)-Y(7,7)	Y(0,7)-Y(7,7)	Y(0,7)-Y(7,7)	Y(0,7)-Y(7,7)	Y(0,7)-Y(7,7)	Y(0,7)-Y(7,7)	Y(0,7)-Y(7,7)	Y(0,7)-Y(7,7)	
64	Y(0,8)-Y(7,8)	Y(0,8)-Y(7,8)	Y(0,8)-Y(7,8)	Y(0,8)-Y(7,8)	Y(0,8)-Y(7,8)	Y(0,8)-Y(7,8)	Y(0,8)-Y(7,8)	Y(0,8)-Y(7,8)	
72	Y(0,9)-Y(7,9)	Y(0,9)-Y(7,9)	Y(0,9)-Y(7,9)	Y(0,9)-Y(7,9)	Y(0,9)-Y(7,9)	Y(0,9)-Y(7,9)	Y(0,9)-Y(7,9)	Y(0,9)-Y(7,9)	
80	Y(0,10)-Y(7,10)	Y(0,10)-Y(7,10)	Y(0,10)-Y(7,10)	Y(0,10)-Y(7,10)	Y(0,10)-Y(7,10)	Y(0,10)-Y(7,10)	Y(0,10)-Y(7,10)	Y(0,10)-Y(7,10)	
88	Y(0,11)-Y(7,11)	Y(0,11)-Y(7,11)	Y(0,11)-Y(7,11)	Y(0,11)-Y(7,11)	Y(0,11)-Y(7,11)	Y(0,11)-Y(7,11)	Y(0,11)-Y(7,11)	Y(0,11)-Y(7,11)	
96	Y(2,0)-Y(9,0)	Y(2,0)-Y(9,0)	Y(2,0)-Y(9,0)	Y(2,0)-Y(9,0)	Y(2,0)-Y(9,0)	Y(2,0)-Y(9,0)	Y(2,0)-Y(9,0)	Y(2,0)-Y(9,0)	
104	Y(2,1)-Y(9,1)	Y(2,1)-Y(9,1)	Y(2,1)-Y(9,1)	Y(2,1)-Y(9,1)	Y(2,1)-Y(9,1)	Y(2,1)-Y(9,1)	Y(2,1)-Y(9,1)	Y(2,1)-Y(9,1)	
112	Y(2,2)-Y(9,2)	Y(2,2)-Y(9,2)	Y(2,2)-Y(9,2)	Y(2,2)-Y(9,2)	Y(2,2)-Y(9,2)	Y(2,2)-Y(9,2)	Y(2,2)-Y(9,2)	Y(2,2)-Y(9,2)	
120	Y(2,3)-Y(9,3)	Y(2,3)-Y(9,3)	Y(2,3)-Y(9,3)	Y(2,3)-Y(9,3)	Y(2,3)-Y(9,3)	Y(2,3)-Y(9,3)	Y(2,3)-Y(9,3)	Y(2,3)-Y(9,3)	
128	Y(2,4)-Y(9,4)	Y(2,4)-Y(9,4)	Y(2,4)-Y(9,4)	Y(2,4)-Y(9,4)	Y(2,4)-Y(9,4)	Y(2,4)-Y(9,4)	Y(2,4)-Y(9,4)	Y(2,4)-Y(9,4)	
136	Y(2,5)-Y(9,5)	Y(2,5)-Y(9,5)	Y(2,5)-Y(9,5)	Y(2,5)-Y(9,5)	Y(2,5)-Y(9,5)	Y(2,5)-Y(9,5)	Y(2,5)-Y(9,5)	Y(2,5)-Y(9,5)	

marked : for Accumulations unmarked : for Y Shifting

Figure 6. The 1-D systolic architecture and the corresponding computing sequence for step 2 of 3HSA



time step n	PE0	PE1	PE2	PE3	PE4	PE5	PE6	PE7	M
	n ~ n+7	n+1~n+8	n+2~n+9	n+3~n+10	n+4~n+11	n+5~n+12	n+6~n+13	n+7~n+14	n
0	Y(0,0)-Y(7,0)	Y(0,0)-Y(7,0)	Y(0,0)-Y(7,0)	Y(0,0)-Y(7,0)	Y(0,0)-Y(7,0)	Y(0,0)-Y(7,0)	Y(0,0)-Y(7,0)	Y(0,0)-Y(7,0)	
8	Y(0,1)-Y(7,1)	Y(0,1)-Y(7,1)	Y(0,1)-Y(7,1)	Y(0,1)-Y(7,1)	Y(0,1)-Y(7,1)	Y(0,1)-Y(7,1)	Y(0,1)-Y(7,1)	Y(0,1)-Y(7,1)	
16	Y(0,2)-Y(7,2)	Y(0,2)-Y(7,2)	Y(0,2)-Y(7,2)	Y(0,2)-Y(7,2)	Y(0,2)-Y(7,2)	Y(0,2)-Y(7,2)	Y(0,2)-Y(7,2)	Y(0,2)-Y(7,2)	
24	Y(0,3)-Y(7,3)	Y(0,3)-Y(7,3)	Y(0,3)-Y(7,3)	Y(0,3)-Y(7,3)	Y(0,3)-Y(7,3)	Y(0,3)-Y(7,3)	Y(0,3)-Y(7,3)	Y(0,3)-Y(7,3)	
32	Y(0,4)-Y(7,4)	Y(0,4)-Y(7,4)	Y(0,4)-Y(7,4)	Y(0,4)-Y(7,4)	Y(0,4)-Y(7,4)	Y(0,4)-Y(7,4)	Y(0,4)-Y(7,4)	Y(0,4)-Y(7,4)	
40	Y(0,5)-Y(7,5)	Y(0,5)-Y(7,5)	Y(0,5)-Y(7,5)	Y(0,5)-Y(7,5)	Y(0,5)-Y(7,5)	Y(0,5)-Y(7,5)	Y(0,5)-Y(7,5)	Y(0,5)-Y(7,5)	
48	Y(0,6)-Y(7,6)	Y(0,6)-Y(7,6)	Y(0,6)-Y(7,6)	Y(0,6)-Y(7,6)	Y(0,6)-Y(7,6)	Y(0,6)-Y(7,6)	Y(0,6)-Y(7,6)	Y(0,6)-Y(7,6)	
56	Y(0,7)-Y(7,7)	Y(0,7)-Y(7,7)	Y(0,7)-Y(7,7)	Y(0,7)-Y(7,7)	Y(0,7)-Y(7,7)	Y(0,7)-Y(7,7)	Y(0,7)-Y(7,7)	Y(0,7)-Y(7,7)	
64	Y(0,8)-Y(7,8)	Y(0,8)-Y(7,8)	Y(0,8)-Y(7,8)	Y(0,8)-Y(7,8)	Y(0,8)-Y(7,8)	Y(0,8)-Y(7,8)	Y(0,8)-Y(7,8)	Y(0,8)-Y(7,8)	
72	Y(0,9)-Y(7,9)	Y(0,9)-Y(7,9)	Y(0,9)-Y(7,9)	Y(0,9)-Y(7,9)	Y(0,9)-Y(7,9)	Y(0,9)-Y(7,9)	Y(0,9)-Y(7,9)	Y(0,9)-Y(7,9)	
80	Y(1,0)-Y(8,0)	Y(1,0)-Y(8,0)	Y(1,0)-Y(8,0)	Y(1,0)-Y(8,0)	Y(1,0)-Y(8,0)	Y(1,0)-Y(8,0)	Y(1,0)-Y(8,0)	Y(1,0)-Y(8,0)	
88	Y(1,1)-Y(8,1)	Y(1,1)-Y(8,1)	Y(1,1)-Y(8,1)	Y(1,1)-Y(8,1)	Y(1,1)-Y(8,1)	Y(1,1)-Y(8,1)	Y(1,1)-Y(8,1)	Y(1,1)-Y(8,1)	
96	Y(1,2)-Y(8,2)	Y(1,2)-Y(8,2)	Y(1,2)-Y(8,2)	Y(1,2)-Y(8,2)	Y(1,2)-Y(8,2)	Y(1,2)-Y(8,2)	Y(1,2)-Y(8,2)	Y(1,2)-Y(8,2)	
104	Y(1,3)-Y(8,3)	Y(1,3)-Y(8,3)	Y(1,3)-Y(8,3)	Y(1,3)-Y(8,3)	Y(1,3)-Y(8,3)	Y(1,3)-Y(8,3)	Y(1,3)-Y(8,3)	Y(1,3)-Y(8,3)	
112	Y(1,4)-Y(8,4)	Y(1,4)-Y(8,4)	Y(1,4)-Y(8,4)	Y(1,4)-Y(8,4)	Y(1,4)-Y(8,4)	Y(1,4)-Y(8,4)	Y(1,4)-Y(8,4)	Y(1,4)-Y(8,4)	
120	Y(1,5)-Y(8,5)	Y(1,5)-Y(8,5)	Y(1,5)-Y(8,5)	Y(1,5)-Y(8,5)	Y(1,5)-Y(8,5)	Y(1,5)-Y(8,5)	Y(1,5)-Y(8,5)	Y(1,5)-Y(8,5)	
128	Y(1,6)-Y(8,6)	Y(1,6)-Y(8,6)	Y(1,6)-Y(8,6)	Y(1,6)-Y(8,6)	Y(1,6)-Y(8,6)	Y(1,6)-Y(8,6)	Y(1,6)-Y(8,6)	Y(1,6)-Y(8,6)	
136	Y(1,7)-Y(8,7)	Y(1,7)-Y(8,7)	Y(1,7)-Y(8,7)	Y(1,7)-Y(8,7)	Y(1,7)-Y(8,7)	Y(1,7)-Y(8,7)	Y(1,7)-Y(8,7)	Y(1,7)-Y(8,7)	

marked : for Accumulations unmarked : for Y Shifting

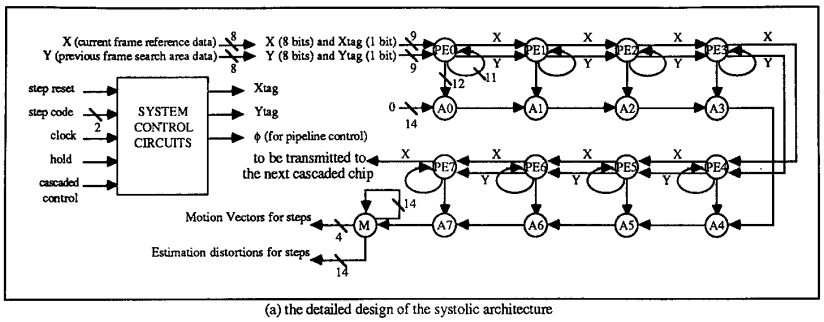
Figure 7. The 1-D systolic architecture and the corresponding computing sequence for step 3 of 3HSA

$$\frac{(9N^2+45N)T_cXY}{N^2} \le \frac{1}{R} \quad (1)$$

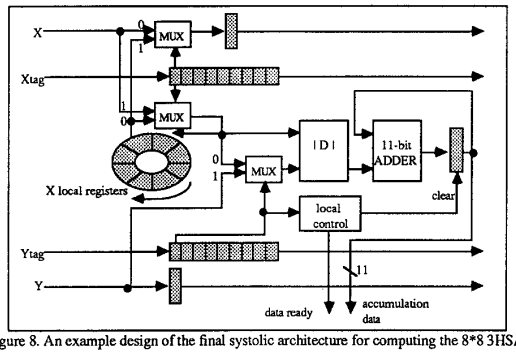
$$\text{therefore, } \frac{N}{XYR(9N+45)} \ge T_c \quad (2)$$

In case the teleconferencing video format (288×352 frame size, 16×16 block size, and 30 Hz frame rate), the constraint is $T_c \le 27.84$ ns. This speed is available in the current VLSI technologies. In other words, the proposed design is practical in the real-world design. On the other hand, in the cascaded system, the throughput required to execute N×N 3HSA is specially limited by step 1 $(3(N+8)NT_c)$. By applying the same video parameters to this cascaded system, we can also obtain a constraint as follows:

$$\frac{N}{XYR(3N+24)} \ge T_c \quad (3)$$

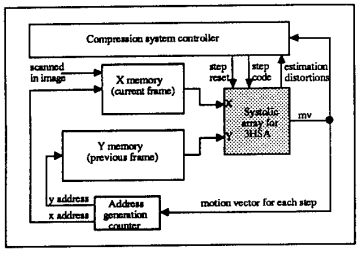


(a) the detailed design of the systolic architecture

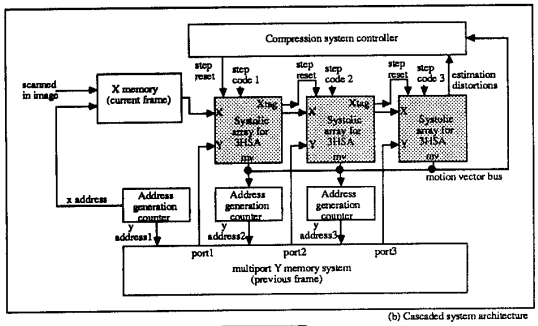


(b) the detailed PE design

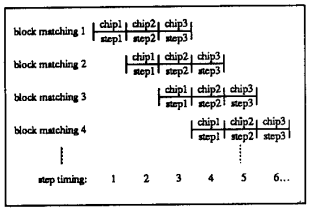
Figure 8. An example design of the final systolic architecture for computing the 8*8 3HSA



(a) system architecture by using single-chip systolic array.



(b) Cascaded system architecture



(c) Time scheduling for cascaded computing

Figure 9. System architecture for 3HSA computing by using the final systolic architecture

In case the broadcast video format (576×720 frame size, 16×16 block size, and 25 Hz frame rate) is applied to (3), the constraint indicates $T_c \leq 21.43$ ns. This rate is also reasonable in the current VLSI technologies. Therefore, both of the above two system architectures are realizable in computing the 3HSA with the current circuit-design technologies.

IV. Conclusion

In real-time video applications, systolic architectures are ideal candidates for VLSI realization. This paper discribes a practical design techniques to realize the systolic architecture for 3HSA. The proposed architecture has the following features: (i) unified execution for each step of 3HSA, (ii) low latency delay, (iii) low I/O bandwidth, (iv) regular hardware structure, and (v) single-chip or cascaded configurations. With the above features, this architecture is very suitable for the VLSI implementation to match the real-time video applications.

REFERENCES

1. Arun N. Netravali and Barry G. Haskell, Digital Pictures Representation and Compression, Plenum, 1988.
2. T. Komarek and P. Pirsch, "Array Architectures for Block Matching Algorithms," IEEE Trans. Circuits and Systems, vol. CAS-36, No. 10, pp 1301-1308, Oct. 1989.
3. K. M. Yang, M. T. Sun, and L. Wu, "A Family of VLSI Designs for the Motion Compensation Block-Matching Algorithm," IEEE Trans. Circuits and Systems, vol. CAS-36, No. 10, pp 1317-1325, Oct. 1989.
4. A. Artieri and F. Jutand, "A Versatile and Powerfull Chip for Real Time Motion Estimation," Proc. IEEE ICASSP , pp. 2453-2456, 1989.
5. S. Y. Kung, VLSI Array Processors, Prentice Hall, Englewood Cliffs, N.J. 1988.