The Optimum Location of Delay Latches Between Dynamic Pipeline Stages

Mahmoud Lotfi Anhar[1] and Mohammad Ali Jabraeil Jamali[2]

[1] Islamic Azad University Khoy Branch , Khoy, Iran
MahmoudLotfie@gmail.com
[2] Islamic Azad University Shabestar Branch, Shabestar , Iran
m_jamali@itrc.ac.ir

Abstract. Latches are used between pipeline stages to get Minimum Average Latency (MAL). An optimization technique based on introducing a method to search the most proper location of noncompute delay latches between nonlinear pipeline stages is given. The idea is to find a new collision vector which is adaptable with pipeline topology and modifies reservation table, yielding MAL at minimum execution time. This approach not only reduces execution time of hardware, but also minimizes favorite collision vector search time.

Keywords: Latch, Pipeline, Minimum, Average, Latency, MAL, Reservation, Table, Collision.

1 Introduction

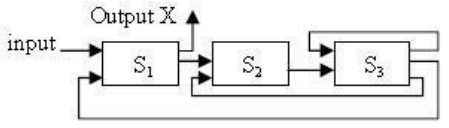
When scheduling events in a pipeline, the main objective is to obtain the shortest average latency between initiations without causing collisions. The pipeline utilization is limited by the collision characteristics which are the result of the usage patterns of the segments. One way of modifying usage pattern is by segment replication. Another way of changing a usage pattern is by inserting noncompute segments, which simply provide a fixed delay between some computation steps. A methodology has presented for modifying the collision characteristics with the insertion of delays so as to increase the utilization of segments by Davidson [1]. The purpose of delay insertion is to modify the reservation table, yielding a new collision vector. This leads to a modified state diagram, which may produce greedy cycles meeting the lower bound the MAL.

2 Single Function Pipelines

The combined set of permissible and forbidden latencies can be displayed by a j-bit binary vector C=($C_jC_{j-1}...C_1$). The value C_i=1 if latency i causes a collision and C_i=0 if latency i is permissible. Figure 1 shows an example of 3 stage pipeline and corresponding reservation table.

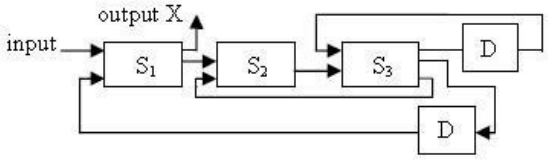
From reservation table MAL must be 2 but state diagram shows 3. Hence delay latches must be used. A solution proposed by Hwang [2]. Two latches are

M. Xu et al. (Eds.): APPT 2007, LNCS 4847, pp. 14–17, 2007.
© Springer-Verlag Berlin Heidelberg 2007



	Time				
	1	2	3	4	5
S_1	X				X
S_2		X		X	
S_3			X	X	

Fig. 1. A 3-stage pipeline and reservation table



	1	2	3	4	5	6	7
S_1	X						X
S_2		X		X			
S_3			X		X		
D				X			
D						X	

Fig. 2. Delay Insertion

added between S3-S3 and S3-S1 feedback loops and modified reservation table is presented in Fig. 2.

Our approach is shown in Fig. 3. It is 1 cycle faster than previous method.

3 Methodology

Assume a j-bit initial collision vector $X_j=(x_j\ x_{j-1}...x_1)$ which must be evaluated to produce {MAL} and $X_i=(x_i\ x_{i-1}...x_1)$ is initial collision vector before inserting latches . First we try evaluate X_j supposing it has a constant latency greedy cycle (G.C) equals the MAL.

Lemma 1. *If G.C=$\{k\}$ then $j \neq q.k$ where q is an integer . Since $x_j = 1$ and SHR[X_j , k] OR $X_j = X_j$, hence $x_k = x_{2k} = ... = 0$,and $j \neq \{k,2k,3k,...\}$.*

Time reduction considerations lead us to minimize j as it is possible. If $i+1 \neq q.k$ we start j with i+1 which is the best choice and yields minimum processing time else j= i+2 is the next choice.

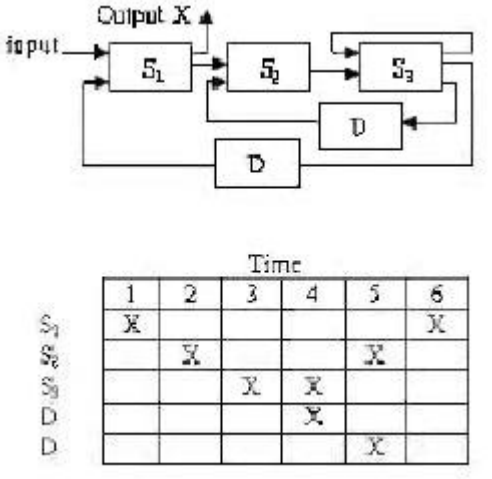


Fig. 3. Insertion delay latches using new approach

Example 1. In Fig. 1 X_i = (1011) and MAL is 2 then we can select j = 5. The new collision vector after inserting latches is X_j = (x_5 x_4 x_3 x_2x_1). Assume at this case MAL is 2, then

SHR[X_j , 2] OR X_j = X_j $\Longrightarrow$ $x_1 = x_3 = x_5 = 1$, $x_2 = x_4 = 0$.

Now examine X_j = (10101). Fig. 3 shows modified state diagram with a reduced MAL. It has better time response than previous approach.

Example 2. Fig. 3 is the reservation table which is used in [1]. Choosing X_j = $(x_7x_6x_5x_4x_3x_2x_1)$, G.C = {3} and using lemma 1 yields X_j= (1011011) and it is a possible solution.

Another way to find X_j is choosing a greedy cycle such as {k_1,k_2}.

Lemma 2. *If G.C = {k_1, k_2} then x_{k1}= x_{k2} = x_{k1+k2}= 0 and we can find two internal states (Y , Z)in state diagram satisfy following conditions:*

1. Y_{K1}=Z_{K2}=0.
2. Z= X OR SHR [Y,k_1].
3. Y=X OR SHR [Z,k_2].

Assume X= (x_j x_{j-1}... x_1) , Y=(Y_j Y_{j-1}...Y_1) and Z=(Z_j Z_{j-1}... Z_1) then
$Z_{j-(k1+i)}$ = $X_{j-(k1+i)}$ OR Y_{j-i+1} , i = 1 to j-(k_1+1).
$Y_{j-(k2+i)}$ = $X_{j-(k2+i)}$ OR Z_{j-i+1} , i = 1 to j-(k_2+1).
Z_{j-k1+i} = X_{j-k1+i} , i = 0 to k_1.
Y_{j-k2+i}= X_{j-k2+i} , i = 0 to k_2.
X_j=Y_j=Z_j= 1.
$Z_{j-(k1+1)}$= 1 , $Y_{j-(k2+1)}$=1
Then j-(k_1+1) $\neq$ k_2 and j-(k_2+1) $\neq$ k_1. j must satisfy following criteria :
j $\neq$ k_1+k_2+1 $\Rightarrow$ j$\geq$ k_1+k_2+2.

Example 3. Another solution to example1 is to choose greedy cycle {1,3}. Finding this greedy cycle is easy by introduced method. First select j = 1+3+2 =6 , then X = (1 x_5 0 0 x_2 0). Now use lemma 2 and find result : X = (1 0 0 0 1 0) , Y = (1 0 0 1 1 0) and Z=(1 1 0 0 1 1). Reservation table of new initial collision vector has 7 columns (1 cycle slower than previous X resulted with {2} greedy cycle).

In this case MAL = $(k_1 + k_2)/2$ then X must be calculated for following set :1, k_1+k_2-1, 2, k_1+k_2-2 , ,$(k_1+k_2)/2$ -1,$(k_1+k_2)/2$ + 1. All of these choices have MAL = $(k_1+k_2)/2$ and number of them is $(k_1+k_2)/2$ - 1. Then we calculate X for constant greedy cycle MAL first, and if it is not sufficient, propose one of above introduced non-constant MALs.

4 Algorithm

The following algorithm is used in this method:

1. For each MAL generate a j-bit collision vector X_j which $j = i + 1$ (i is the number of initial collision vector X_i bits). Assume greedy cycle is MAL.
2. Test X . If X satisfies then quit, else $j = j + 1$;
3. Try 2 until j < k_1+k_2+2 .
4. Select one of non-constant previously stated greedy cycles. Select another greedy cycle if does not satisfy criteria stated in lemma 2.

5 Concluding Remarks

We have presented a new method to evaluate initial collision vector using constant and non-constant greedy cycles. For increasing throughput and then reducing processing time we began with constant greedy cycles. Then other non-constant cycles related to MAL is used. This method not only reduces processing time but also is a faster one than other methods.

References

1. Patel, J.H., Davidson, E.S.: Improving the Throughput of a Pipeline by Insertion of Delays. Coordinated Science Lab, University of Illinois, Urbana, Illinois 61801, 132–137 (1976)
2. Hwang, K.: Advanced Computer Architecture: Parallelism, Scalability, Programmability. McGraw-Hill, New York (1993)
3. Davidson, E.S., Thomas, D.P., Shar, L.E., Patel, J.H.: Effective Control for Pipelined Computers. In: Proc. COMPCON, pp. 181–184 (1975)
4. Kogge, P.M.: The Architecture of Pipelined computers. Mc-Graw-Hill, New York (1981)