An Optimal Register resource Allocation Algorithm using Graph Coloring

Ji-young Choi,
Dept. of Electronic Engineering, Chongju University
36 Naedok-Dong Sangdang-Gu, Chongju Chungbuk Korea ,
m7515103@kebi.com ,

Chi-ho Lin*,
*Dept. of Computer Science, Semyung University ,
San 21-1, Shinwol-Dong, Chechon Chungbuk Korea,
ich410@venus.semyung.ac.kr* ,

Hi-seok Kim**
**Dept. of Electronic Engineering, Chongju University
36 Naedok-Dong, Sangdang-Gu, Chongju Chungbuk Korea
khs8391@chongju.ac.kr**

Abstract

This paper proposed an optimal register resource allocation algorithm using graph coloring for minimal register at high-level synthesis.

The proposed algorithm constructed interference graph consist of the intermediated representation CFG to description VHDL. and at interference graph for the minimal select color selected a position node at stack, the next inserted spill code and the graph coloring process executes for optimal register allocation.

The proposed algorithm proves to effect that result compare another allocation techniques through experiments of bench mark.

1. Introduction

The purpose of high-level synthesis will realize behavioral description as hardware resource to the minimum cost function.[1,2] thereby the definition of high level synthesis was consisted by scheduling, allocation, binding from behavioral description of designed to create structure of RT(register-transfer) level for limiting constraints and satisfied target function. The scheduling consist of assigning behavioral description each operation to a control step.[3,4] But it had developed algorithms solution of a limited application field for very scheduling process is considerable items conditional branch, pipeline, loop and so on. The allocation is assigned so that minimize area of implement hardware, to operation as functional unit, to variable as register, between operation and register as interconnection assigned to bus and multiplexer. The memory resource allocation is used to the register. Because data access time is the fastest memory storages, the goal of register allocation minimize delay time and cost function as following memory reference many variable as much as possible.

The existed graph coloring techniques of register allocation , the first implementation of a register allocator was described by Chaitin et al.[5] We assume that all variable of register allocation exist virtual register. the register allocation process composed five step. Namely, live range, interference graph, coalesce, spill and coloring. The spill necessary element for allocating register to limit the number. But spill cause memory reference. If possible, the spill reduction bring up important problem by register allocation for graph coloring.[6]

On the other hand Briggs technique fill up the weakness chaitin technique to change viewpoint of the spill according to execute spill about coloring is possible like chaitin technique. It is apt to execute an effective register allocation with high performance the minimum spill but have same function it f-or using that instruction for memory access is bigger execution time than general instruction through one computation again solving the value the spill isn't execute to preserve the value as insert spill code through one computation.[7] This paper graph coloring was based on Briggs technique.

The structure of this paper was as following. The section 2 describes an optimal register allocation algorithm using graph coloring. The section 3 improve effective paper through experiment and consideration. The end section 4 is composed of the conclusion.

2. An optimal register resource allocation algorithm using graph coloring

The graph coloring is painted each node in the color that two node connected edge in graph is another color. Each node oh the graph assign variable. If it connected that each other register allocated as variable belong the edge between node. The problem which painted the color each node is able to treat the problem that register allocate each variable.

In generally, an issue painted graph G as the K color become known NP-complete. The edge under degree K in interf-

erence graph of the node n that another node how was regard less of coloring always is able to paint.

So, to solve the problem using graph coloring use optimal technique. The block diagram of an optimal register allocat ion algorithm using graph coloring shown in Figure 1.

The process of the whole allocation of optimal register allocation algorithm that constructed interference graph consist o f intermediated representation CFG(Control Flow Graph) tod escription VHDL. Suppose, when it assume to use register k in the interference graph, if not the node degree(n) $\leq$ k (n : n ode , k : the usable register number), instead of spilling it sel ect the positional node at stack. The coloring assume usi-ng c olor when the node pop in the stack. If the color not useable, when the node pop in the stack. That is, suppose degree(n) > k, the node not painted and it execute continually coloring. I f not color node consist in the end of coloring process, it ins- ert necessary spill code and interference graph reorganized.

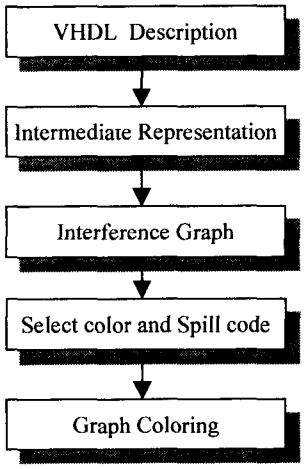

Fig. 1. An optimal register resource allocation algorithm

2.1 The VHDL description

It inputted VHDL (Very High speed Integrated Circuit Ha- rdware description Language) for accomplishing an optimal register allocation algorithm. The VHDL is language usable for description hardware behavioral to user. The representation method of VHDL divide by behavioral modeling, data- flow modeling, and structure modeling. This paper example of an optimal register resource allocation algorithm for using graph coloring is based on behavioral modeling. The behav- ioral modeling describe the system function using functional or mathematical algorithm to want designer regardless of the internal some structure from the highest abstractly represent- ation among the three model. An example VHDL input shown in Figure 2.

```
entity color_3 is
port ( a, b, c, k : in INTEGER;
       c,e : out INTEGER );
end color_3;
architecture behavioral of color_3 is
 begin
   d := a + c + 2;
   if (a < b) then
    c := a –1 + b;
  else
    c := a + b;
  end if
  e := c * k;
 end behavioral;
```

Fig. 2. The behavioral modeling VHDL input

2.2 The intermediate representation

Through behavioral transformation accept on the input VH DL source file, created CFG(Control Flow Graph) suitable i n high-level synthesis. The intermediate representation is n- ecessary information of the data and control from VHDL for the high-level synthesis. It is the configuration three adders- s. The each real variable is mapping $(number), in this case two operation above, temperate variable (temp) allocated a middle stage.

The high-level synthesizer constitute scheduler and module allocator, the input accept the created CFG for VHDL ana- lyzer. From intermediated representation the created CFG is like figure 3. Also, it divide each four control step as the re- sult scheduling. In particular three control step cause resour ce sharing according to the condition. The fork used for expr ession comparison statement. it is a branch role control flow as condition an expression. The join used when separate br- anch integrate.

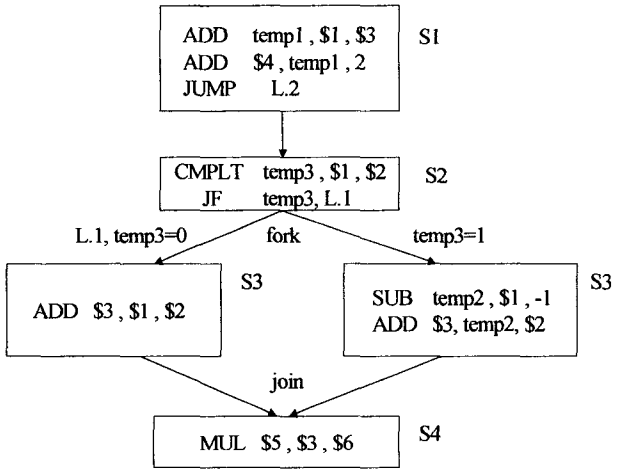

Fig. 3. Intermediate Representation

2.3 The interference graph

The interference graph was defined that it painted k color to allocate register for graph coloring at k register machine. The interference graph shown in Figure 3. The node indica -te lift-time of the variable using CFG and the edge means in terference between life-time.

The definition of the life-time exposes between distance for m the view to define(write) to the view to use(read). That is, it says the live range of variable. The life-time of schedulin g result shown in Figure 4. There is the allocated result each control step. The first control step is allocated $1, $3, $4, t1 a nd two control step is allocated $1, $2, t3. The three control step of resource sharing is allocated $1, $2, $3, t2 and four control step is allocated $3, $5, $6.

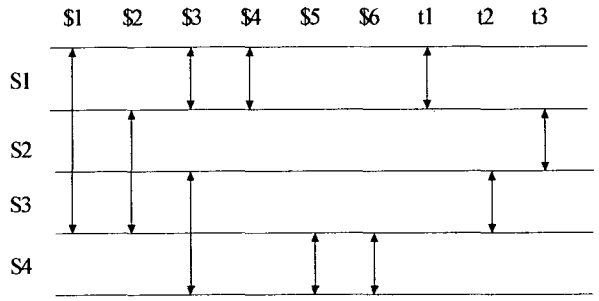

Fig. 4. The life-times

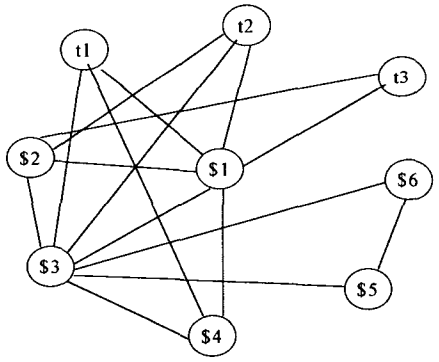

Fig. 5. Interference graph

2.4 The select color and spill code

First of all, if the existence node degree(n) ≤ k push in coloring stack. but though the existence node degree(n) > k not s pilled. If because Chaitin technique degree(n) > k, it is spilled in order as the priority to decide it of spilling variable among variable candidate that express unconditionally spilled variable candidate. If it was using allocated that bring about execution time and wastefulness on the memory. In other side, an optimal register allocation using graph coloring in excess of usable register number, that is, in this case spilled that push coloring stack from the big degree. If the same degree node the virtual register that is, it is storage color stack from the big $(number). The minimum interference graph shown in Table 1.

Table. 1. The minimum interference graph

stack / var	Sam	$5	$6	t3	$1	$4	t1	$3	$2	t2
t1	3	3	3	3	2	1	X	X	X	X
t2	3	3	3	3	2	2	2	1	1	X
t3	2	2	2	X	X	X	X	X	X	X
$1	6	6	6	5	X	X	X	X	X	X
$2	4	4	4	3	2	2	2	1	X	X
$3	7	6	5	5	4	3	2	X	X	X
$4	3	3	3	3	2	X	X	X	X	X
$5	2	X	X	X	X	X	X	X	X	X
$6	2	1	X	X	X	X	X	X	X	X

2.5 The graph coloring

The register assign that painted special color in graph node . The graph reconstructed by inverse of the eliminated order (this order remembered for using stack). This is allocated re gister as the minimum process. However spill code remove the node and not allocated it. Therefore it reconstitute the interference graph. The coloring algorithm shown in Figure 6. The ultimate the interference graph of an optimal register allocation shown in Figure 7 . As you see the figure 7, the regi ster R1(red) is allocated t1, t2, $6 and the register R2(green) is alloc-ated $2, $4, $5. as the same time R3(blue) is allocated t3, $3. Therefore it is obtained the minimum register.

```
if(node) {
    Color_stack_pop( );
    if(degree(n) > k) {
        Non_coloring( );
        Spill_code( );
    else
        Coloring( );
    }
}
```

Fig. 6. Coloring algorithm

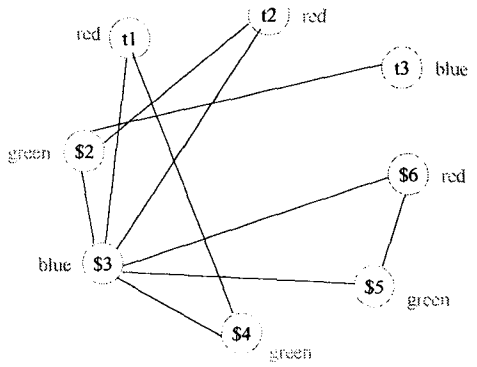

Fig. 7. The final interference graph

3. Experimental results

This paper made an experiment allocation process through st andard benchmark SPEC '95. In the five a fixed number eac h go, gcc, compress, li, vortex. The spilled register shown in Table 2. The execution time is fast even if spilled register is small. The benchmark li is effectiveness which t-he proposed algorithm is 38%. The experiment result towar-d execution time shown in Table 3.

Table. 2. Compare experiment of the spilled register

benchmark	Chaitin's	ours	Pct.
Go	3	3	0
Gcc	5	4	20
compress	8	6	25
Li	13	8	38
vortex	17	11	35

Table. 3. Compare experiment the execution time

benchmark	Chaitin	Ours	Pct.
go	8.2	8.2	0
gcc	8.3	8.2	1
compress	8.7	8.4	3
li	10.0	8.9	11
vortex	13.2	11.2	15

Table 3 shows comparison experiment the execution time. The benchmark go is not change but the compress, li, vortex shows the superiority.

4. Conclusion

This paper proposed an optimal register allocation algorith m using graph coloring for minimal register at high-level sy-nthesis.

The proposed algorithm constructed interference graph con-sist of intermediated representation CFG to description VHD L. and at interference graph for the minimal select color sele-cted a position node at stack, the next inserted spill code.

The spill code is necessary the step of reduction spill code and finally the graph coloring process executes for optimal register allocation. this algorithm reduce, as you see, the spill code maximum 38% and the execution time also 15%.

After this study project will precede the research for execu-ting technique to register allocation and scheduling at the sa-me time and performance estimation for a various benchmar-k.

Reference

[1] D. Gajski, N. Dutt, A. Wu, S. Lin, High-level Synth esis, Kluwer Academic Publishers, 1992.

[2] P. Michel, U. Laughter, P. Duzy, The Synthesis Approac h to Diggital System Design, Kluwer Academic Publishers, 1992.

[3] R. Camposano, "From Behavior to Structure: High-Leve l Synthesis", IEEE Design & Test of Computer, pp.8-19, Oct .1990.

[4] James R. Armstrong, F. Gall Gray." Structured Logic De sign With VHDL" , 1993

[5] Gregory J. Chaitin., "Register allocation and spilling via graph coloring". SIGPLAN Notices,17(6) :98-105, June 198 2. Proceeding of the ACM SIGPLAN'82 Symposium on co mpiler Construction.

[6] D. Bernstein et al. "Spill code minimization techniques fo r optimizing compiler," SIGPLAN Notices 24, Jul, 258-263, 1989.

[7] P. Briggs, "Register Allocation via graph Coloring," Ph. D. dissertation, Univ of Rice, 1992.