Testbench on C2FSMD by Rathan Kumar

Alan Tom Jose(2018A7PS0187G) , Aanand SJ(2018A7PS0139G)

Birla Institute of Technology and Science, Pilani

Goa Campus

This document gives the observations on running various test cases on the C2FSMD tool developed by Rathan Kumar. We performed the tests with various types of simple C code and passed it through the C2FSMD Parser.

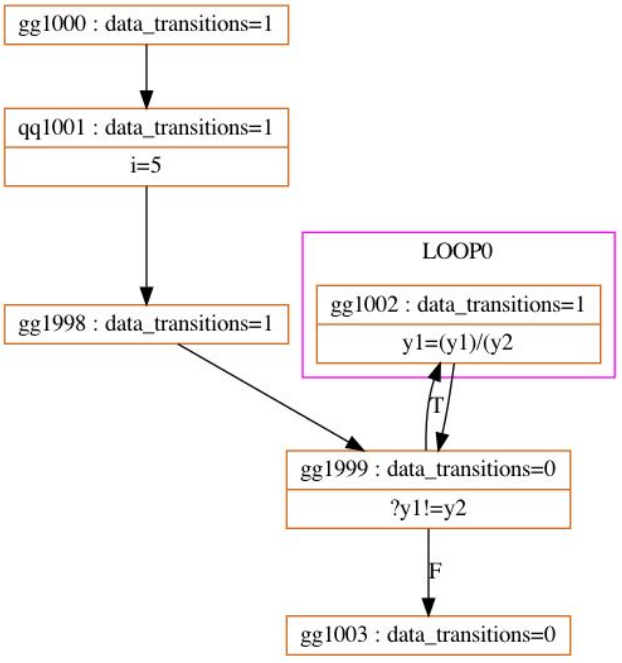
I. Testbench

A. *Program with single while loop*

a. Code

```
#include <stdio.h>
int gcd_modified()
{
   int y1, y2;
   int z;
   int i = 5;
   while (y1 != y2)
   {
       y1 = y1 / y2;
   }
   return z;
}
```

b. Graph



c. Remarks

Works properly without error and generates the correct graph.

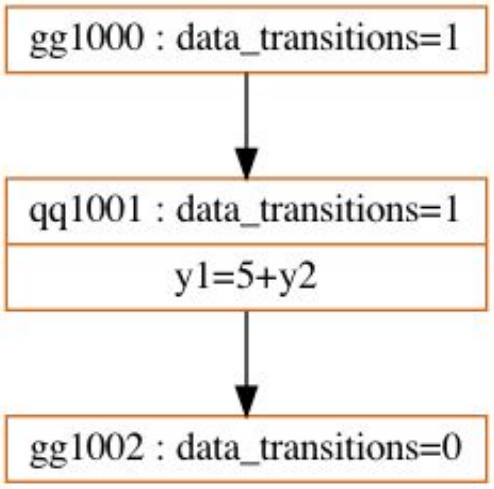
B. *Simple program without loop or if statement*

a. *Code*

```
//test gg

#include <stdio.h>
#include "addg1.h"
#define NOTHING
int test()
{
   int y1, y2;
   y1 = 5 + y2;
   return y1;
}
```

b. *Graph*



c. *Remarks*

Works properly without error and generates the correct graph.

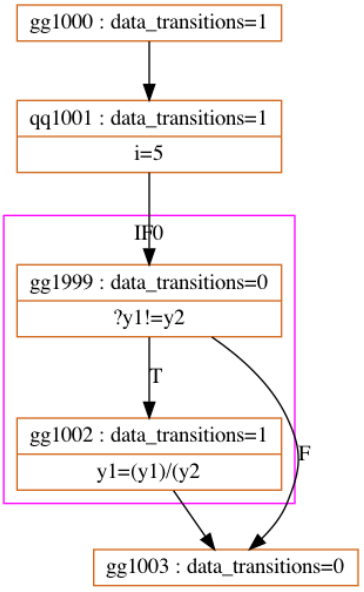
C. Program with no loop and only if.

a. Code

```
//test gg


#include <stdio.h>
int gcd_modified()
{
   int y1, y2;
   int z;
   int i = 5;
   if (y1 != y2)
   {
       y1 = y1 / y2;
   }
   return z;
}
```

b. Graph



c. Remarks

Works properly without error and generates the correct graph.

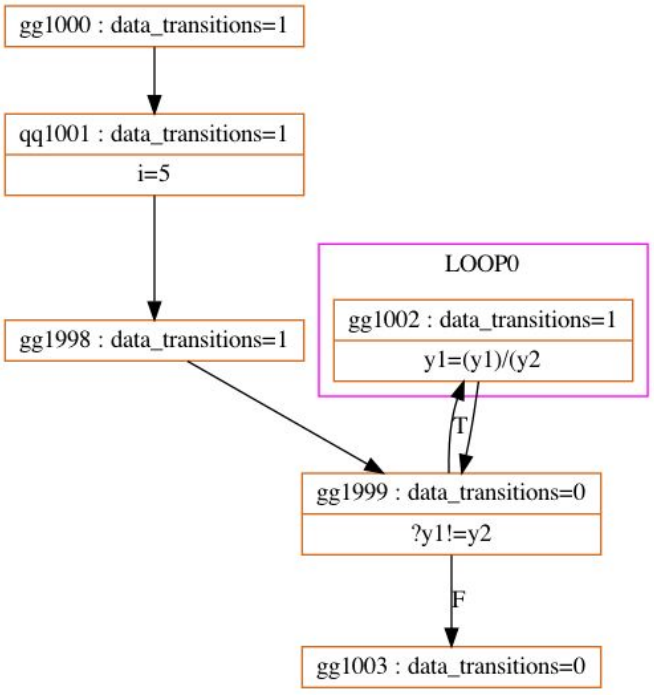
D. Program with do while

a. Code

```
//test gg


#include <stdio.h>
#define NOTHING
int test()
{
   int y1, y2;
   int z;
   int i = 5;
   do
   {
       y1 = y1 / y2;
   } while (y1 != y2);
   return z;
}
```

b. Graph



c. Remarks

Works properly without error and generates the correct graph.

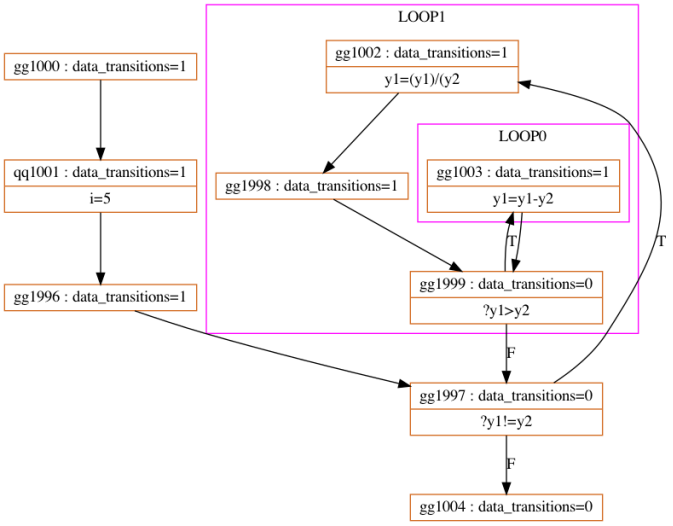
E. Program with Nested Loop

a. Code

```
//test gg


#include <stdio.h>
#define NOTHING
int test()
{
   int y1, y2;
   int z;
   int i = 5;
   while (y1 != y2)
   {
       y1 = y1 / y2;
       while (y1 > y2)
       {
                   y1 = y1 - y2;
       }
   }
   return z;
}
```

b. Graph



c. Remarks

Works properly without error and generates the correct graph.

F. Program with Array

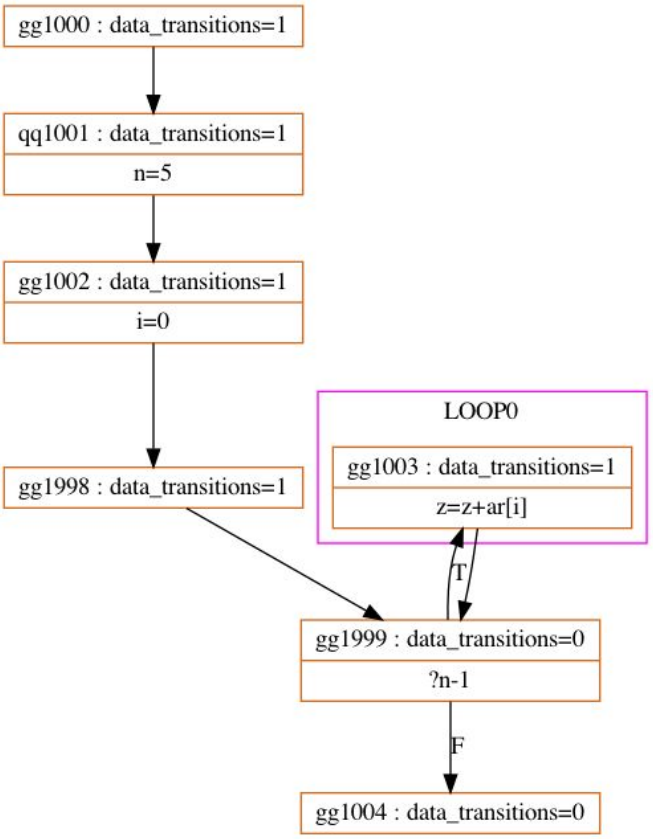
a. Code

```
//test gg

#include <stdio.h>
#define NOTHING
int test()
{
   int n = 5, z;
   int ar[n];
   int i = 0;
   while (--n)
   {
       z += ar[i];
   }

   return z;
}
```

b. Graph



c. Remarks

Generates a graph even when the variable in the array index is not declared in the code.

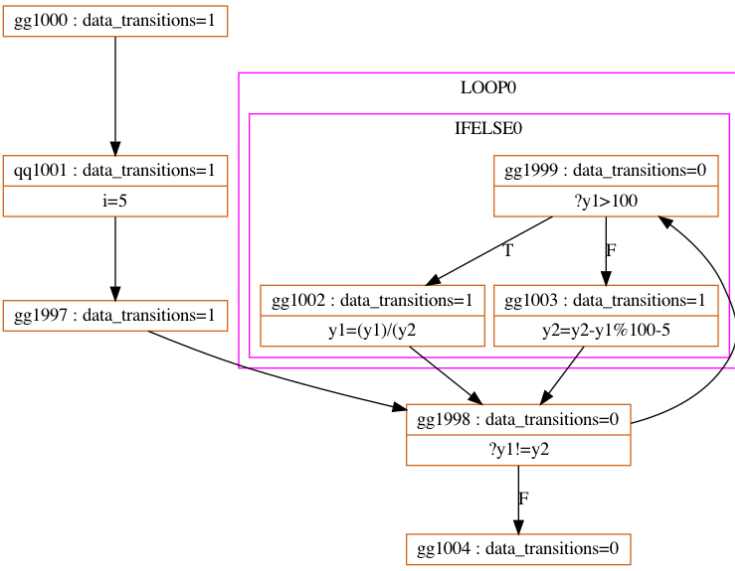
G. Program with If inside While

a. Code

```
//gcd gg


#include <stdio.h>
int gcd_modified()
{
   int y1, y2;
   int z;
   int i = 5;
   while (y1 != y2)
   {
       if (y1 > 100)
                   y1 = y1 / y2;
       else
                   y2 = (y2 - y1) % 100 - 5;
   }
   return z;
}
```

b. Graph



c. Remarks

Works properly without error and generates the correct graph.

H. Program with multiple functions

a. Code

```
//test gg


#include <stdio.h>
#define NOTHING
int test();

int main()
{
   int y1, y2;
   int z;
   z = test();
   while (y1 != y2)
   {
       y1 = y1 / y2;
   }
   return z;
}

int test()
{
   int y1, y2;
   y1 = 5;
   return y1;
}
```

b. Graph

Nil

c. Remarks

Here the tool picks up the function declaration of the test and tries to process the empty declaration and gives segmentation fault.

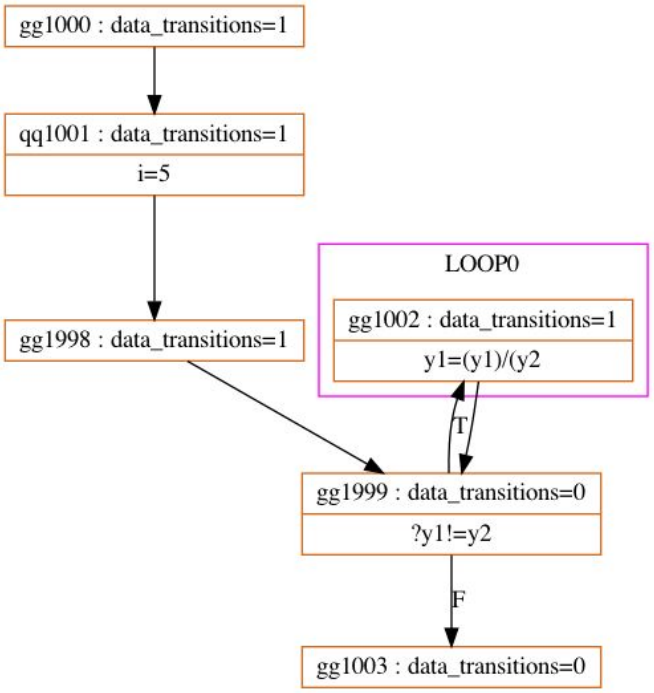
I. Program without proper return

a. *Code*

```
//test gg


#include <stdio.h>
#include "addg1.h"
#define NOTHING
int test()
{
   int y1, y2;
   int z;
   int i = 5;
   while (y1 != y2)
   {
       y1 = y1 / y2;
   }
}
```

b. *Graph*



c. *Remarks*

Should give error as C code does not compile but generates graph nevertheless.

J. Program with expression between variable definition.

a. Code

```
//test gg


#include <stdio.h>


#define NOTHING
int test()
{
   int y1, y2;
   y1 = 5;
   int z;
   while (y1 != y2)
   {
       y1 = y1 / y2;
   }
}
```

b. Graph

Nil

c. Remarks

Gives syntax error even though code does contain any error.

K. Very Large Program.

a. Code

```
//test gg

#include <stdio.h>
#include "addg1.h"
#define NOTHING
int test()
{
   int y1, y2;
   int z, x;
   int i = 5;
   while (y1 != y2)
   {
       if (y1 > 100)
                   y1 = y1 / y2;
       else
                   y2 = (y2 - y1) % 100 - 5;
   }

……. (~600 lines)
……..

while (y1 != y2)
   {
       if (y1 > 100)
                   y1 = y1 / y2;
       else
                   y2 = (y2 - y1) % 100 - 5;
   }
   x = 5;
   return z;
}
```

b. Graph

(Very Large png ~7mb)

c. Remarks

Generates proper output but graph is too big to generate or use effectively

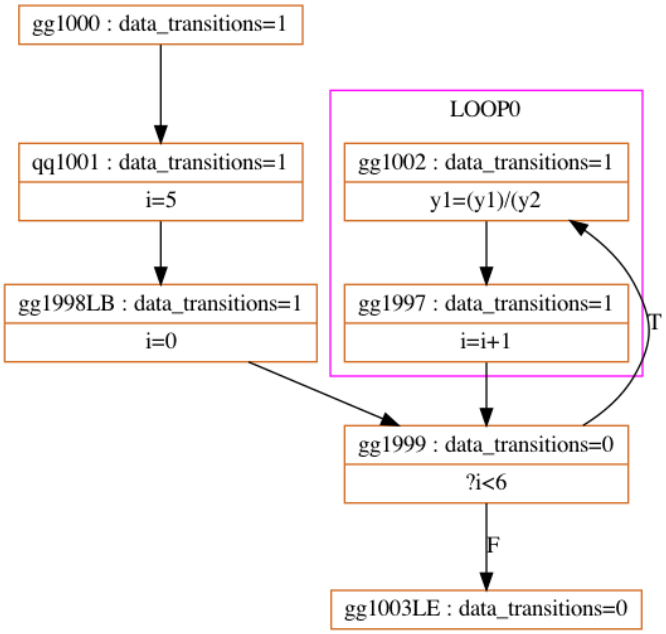
L. Program with for loop.

d. Code

```
//test gg


#include <stdio.h>
#define NOTHING
int test()
{
   int y1, y2;
   int z;
   int i = 5;
   for (i = 0; i < 6; i++)
   {
       y1 = y1 / y2;
   }
   return z;
}
```

e. Graph



f. Remarks

Works properly without error and generates the correct graph. But the code gives error if the iterating variable is declared inside the for loop brackets.

M. Program with dynamic memory allocation.

a. *Code*

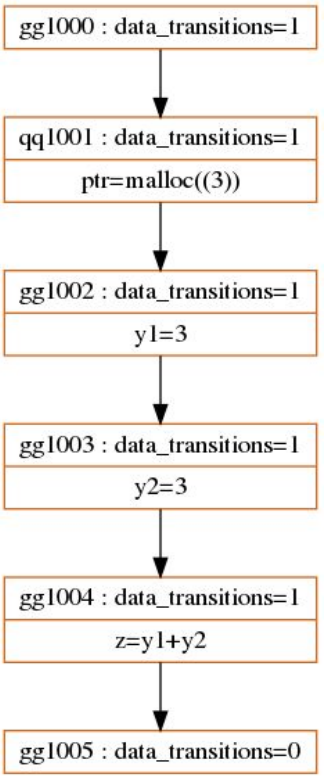
```
//gcd gg

#include <stdio.h>
#include "addg1.h"
#define NOTHING



void gcd()
{
    int y1, y2;
    int z;
    int* ptr=(int*)malloc(3*sizeof(int));
    y1 = 3;
    y2 = 3;
    z = y1 + y2;
}
```

b. *Graph*



c. *Remarks*

Works properly without error and generates the correct graph.

N. Program with structure definition.

a. Code

```
//test gg

#include <stdio.h>
#include <stdlib.h>
#define NOTHING

struct Student
{
   int rollno;
   char name[5];
};

void test()
{
   int n = 5, z;
   struct Student s;
   s.rollno = 5;
   while (--n)
   {
       z += 2;
   }


}
```

b. Graph

Nil

c. Remarks

Gives segmentation fault.

II. Conclusions

From our tests we observed the following conclusions.

1. Programs without loops or if statements and with simple loops or if work properly.
2. For programs with arrays, no error is given even when the variable inside the array brackets is undefined.
3. For programs with multiple functions, only the first function is considered. If it is a function declaration, it is processed and gives segmentation fault.
4. For programs without proper return, graphs are generated and no error is given even though the C code has return error.
5. Gives error if expressions are present in between variable declarations.
6. For very large programs, .org and .dotty files are generated but the .png of graphs made is too big to be processed.
7. If a variable is declared as int and then used as array later in the code, the tool does not catch the error and generates graph properly
8. Tool gives segmentation fault when structures are used in code.
9. Const, double ,enum, extern, float in code gives syntax error.