Addendum

Fully operational software in Halt.c of linked repository conclusively proves:
(a) DD is correctly simulated by HH and the simulated HH(DD,DD) that DD calls.
(b) DD is correctly simulated by HH remains stuck in recursive simulation that cannot possibly terminate normally.

Proof that executed HH(DD,DD) and simulated HH(DD,DD) simulate DD correctly. This proof requires expert knowledge of the C programming language and the x86 programming language. I had to make this proof 100% precise using the x86 language so that false assumptions about the behavior of DD correctly simulated by HH could be unequivocally refuted.

The same thing can be easily seen by any expert in the C programming language. HH(DD,DD) simulates its input that reaches line 03 of DD to invoke another recursive simulation where HH simulates itself simulating DD. No simulated DD ever reaches past its own line 03.

With this expertise it is easy to confirm that both the directly executed HH(DD,DD) and the simulated HH(DD,DD) simulate the steps of DD exactly the way that the x86 machine language specifies. They are simulated correctly and in the correct order.

Lines 01 through 08 of the x86 source code of DD are proven to be correctly simulated by HH and the simulated HH that the simulated DD calls because the execution trace that HH and the simulated HH derives exactly match these lines.

If one also has expertise on the mapping from the C source code to the x86 assembly language then one also confirms that the x86 version of DD is exactly what the C source-code specifies.

```
01   int DD(int (*x)())
02   {
03     int Halt_Status = HH(x, x);
04     if (Halt_Status)
05         HERE: goto HERE;
06     return Halt_Status;
07   }
08
09   int main()
10   {
11     Output("Input_Halts = ", HH(DD,DD));
12   }
```

```
_DD()
[00001db2] 55         push ebp          ; DD line 01
[00001db3] 8bec       mov ebp,esp       ; DD line 02
[00001db5] 51         push ecx          ; DD line 03
[00001db6] 8b4508     mov eax,[ebp+08]  ; DD line 04 param
[00001db9] 50         push eax          ; DD line 05 push param
[00001dba] 8b4d08     mov ecx,[ebp+08]  ; DD line 06 param
[00001dbd] 51         push ecx          ; DD line 07 push param
[00001dbe] e8bff5ffff call 00001382     ; DD line 08 call HH
[00001dc3] 83c408     add esp,+08
[00001dc6] 8945fc     mov [ebp-04],eax
[00001dc9] 837dfc00   cmp dword [ebp-04],+00
[00001dcd] 7402       jz 00001dd1
[00001dcf] ebfe       jmp 00001dcf
[00001dd1] 8b45fc     mov eax,[ebp-04]
[00001dd4] 8be5       mov esp,ebp
[00001dd6] 5d         pop ebp
[00001dd7] c3         ret
Size in bytes:(0038) [00001dd7]


 machine   stack     stack     machine    assembly
 address   address   data      code       language
 ========  ========  ========  =========  =============
[00001de2][00103292][00000000] 55         push ebp
[00001de3][00103292][00000000] 8bec       mov ebp,esp
[00001de5][0010328e][00001db2] 68b21d0000 push 00001db2
[00001dea][0010328a][00001db2] 68b21d0000 push 00001db2
[00001def][00103286][00001df4] e88ef5ffff call 00001382
New slave_stack at:103336
Begin Local Halt Decider Simulation   Execution Trace Stored at:11333e
[00001db2][0011332a][0011332e] 55         push ebp          ; DD line 01
[00001db3][0011332a][0011332e] 8bec       mov ebp,esp       ; DD line 02
[00001db5][00113326][001032fa] 51         push ecx          ; DD line 03
[00001db6][00113326][001032fa] 8b4508     mov eax,[ebp+08]  ; DD line 04
[00001db9][00113322][00001db2] 50         push eax          ; push DD
[00001dba][00113322][00001db2] 8b4d08     mov ecx,[ebp+08]  ; DD line 06
[00001dbd][0011331e][00001db2] 51         push ecx          ; push DD
[00001dbe][0011331a][00001dc3] e8bff5ffff call 00001382     ; call HH
New slave_stack at:14dd5e
[00001db2][0015dd52][0015dd56] 55         push ebp          ; DD line 01
[00001db3][0015dd52][0015dd56] 8bec       mov ebp,esp       ; DD line 02
[00001db5][0015dd4e][0014dd22] 51         push ecx          ; DD line 03
[00001db6][0015dd4e][0014dd22] 8b4508     mov eax,[ebp+08]  ; DD line 04
[00001db9][0015dd4a][00001db2] 50         push eax          ; push DD
[00001dba][0015dd4a][00001db2] 8b4d08     mov ecx,[ebp+08]  ; DD line 06
[00001dbd][0015dd46][00001db2] 51         push ecx          ; push DD
[00001dbe][0015dd42][00001dc3] e8bff5ffff call 00001382     ; call HH
Local Halt Decider: Recursive Simulation Detected Simulation Stopped

[00001df4][00103292][00000000] 83c408     add esp,+08
[00001df7][0010328e][00000000] 50         push eax
[00001df8][0010328a][00000743] 6843070000 push 00000743
[00001dfd][0010328a][00000743] e8a0e9ffff call 000007a2
Input_Halts = 0
[00001e02][00103292][00000000] 83c408     add esp,+08
[00001e05][00103292][00000000] eb79       jmp 00001e80
[00001e80][00103292][00000000] 33c0       xor eax,eax
[00001e82][00103296][00000018] 5d         pop ebp
[00001e83][0010329a][00000000] c3         ret
Number of Instructions Executed(16829) == 251 Pages
```

```
_main()
[00001de2] 55          push ebp
[00001de3] 8bec        mov ebp,esp
[00001de5] 68b21d0000  push 00001db2
[00001dea] 68b21d0000  push 00001db2
[00001def] e88ef5ffff  call 00001382
[00001df4] 83c408      add esp,+08
[00001df7] 50          push eax
[00001df8] 6843070000  push 00000743
[00001dfd] e8a0e9ffff  call 000007a2
[00001e02] 83c408      add esp,+08
[00001e05] eb79        jmp 00001e80
[00001e07] 6a05        push +05
[00001e09] 68321d0000  push 00001d32
[00001e0e] e84ff7ffff  call 00001562
[00001e13] 83c408      add esp,+08
[00001e16] 50          push eax
[00001e17] 6853070000  push 00000753
[00001e1c] e881e9ffff  call 000007a2
[00001e21] 83c408      add esp,+08
[00001e24] 68621d0000  push 00001d62
[00001e29] e834faffff  call 00001862
[00001e2e] 83c404      add esp,+04
[00001e31] 50          push eax
[00001e32] 6863070000  push 00000763
[00001e37] e866e9ffff  call 000007a2
[00001e3c] 83c408      add esp,+08
[00001e3f] 6a05        push +05
[00001e41] 68f21c0000  push 00001cf2
[00001e46] e817f7ffff  call 00001562
[00001e4b] 83c408      add esp,+08
[00001e4e] 50          push eax
[00001e4f] 6873070000  push 00000773
[00001e54] e849e9ffff  call 000007a2
[00001e59] 83c408      add esp,+08
[00001e5c] 68821d0000  push 00001d82
[00001e61] 68821d0000  push 00001d82
[00001e66] e8f7f6ffff  call 00001562
[00001e6b] 83c408      add esp,+08
[00001e6e] 50          push eax
[00001e6f] 6883070000  push 00000783
[00001e74] e829e9ffff  call 000007a2
[00001e79] 83c408      add esp,+08
[00001e7c] 33c0        xor eax,eax
[00001e7e] eb02        jmp 00001e82
[00001e80] 33c0        xor eax,eax
[00001e82] 5d          pop ebp
[00001e83] c3          ret
Size in bytes:(0162) [00001e83]
```