Program Counter Testing

by William Harrington

Introduction

The purpose of this document is to examine whether or not the program counter operates as intended.

Design

The program counter is designed to only increment when the Enable (EN) signal goes high. It should increment by 4 every time this occurs. The ability of the program counter to jump is facilitated by the Load signal. Whenever Load is asserted, whatever value is on the Address signal (addr) is the next output of the program counter.

Testing

Need to examine the following:

- PC only increments when EN asserted
- PC increments by 4
- When load asserted PC output is addr

Description

The testbench in testbench.py is called 5 separate times to observe the functionality described in the previous sections. In the testbench, the enable signal is asserted every 20ns. On the positive edge of the enable signal, the stimulus checks to see if the clock cycle count is greater than 5 and calls the randrange function with an argument of 2. The randrange function returns either 0 or 1. If it returns 1 and the clock cycle count is greater than 5, then the load signal is asserted and a random number between 0 and 16 that divides by 4 evenly is generated then placed on the addr signal. For more information, see testbench.py.

Test 1

In [1]:

```
from testbench import *
main()
```

```
t(ns)    cycle   output  addr    en  load
-----------------------------------------------
20   0   0x0L    0x0L    1   0
60   2   0x4L    0x0L    1   0
100  4   0x8L    0x0L    1   0
140  6   0xcL    0x0L    1   0
180  8   0x10L   0x0L    1   0
220  10  0x0L    0x0L    1   1
260  12  0x4L    0x0L    1   0
300  14  0x8L    0x0L    1   0
340  16  0x0L    0x0L    1   1
380  18  0x0L    0xcL    1   1
420  20  0xcL    0x4L    1   1
460  22  0x4L    0xcL    1   1
500  24  0xcL    0x0L    1   1
540  26  0x0L    0x8L    1   1
580  28  0x8L    0x4L    1   1
620  30  0x8L    0x0L    1   0
660  32  0xcL    0x0L    1   0
700  34  0x10L   0x0L    1   0
740  36  0x0L    0x0L    1   1
780  38  0x4L    0x0L    1   0
820  40  0x4L    0x4L    1   1
860  42  0x8L    0x0L    1   0
900  44  0xcL    0x0L    1   0
940  46  0xcL    0xcL    1   1
980  48  0x10L   0x0L    1   0
<class 'myhdl._SuspendSimulation'>: Simulated 1000 timesteps
```

Test 2

In [1]:

```
from testbench import *
main()
```

```
t(ns)    cycle   output  addr    en  load
-----------------------------------------------
20   0   0x0L    0x0L    1   0
60   2   0x4L    0x0L    1   0
100  4   0x8L    0x0L    1   0
```

```
140  6   0xcL    0x0L    1   0
180  8   0x10L   0x0L    1   0
220  10  0x14L   0x0L    1   0
260  12  0xcL    0xcL    1   1
300  14  0xcL    0x0L    1   1
340  16  0x0L    0xcL    1   1
380  18  0xcL    0x0L    1   1
420  20  0x0L    0xcL    1   1
460  22  0xcL    0x8L    1   1
500  24  0x8L    0xcL    1   1
540  26  0x10L   0x0L    1   0
580  28  0x14L   0x0L    1   0
620  30  0x8L    0x8L    1   1
660  32  0xcL    0x0L    1   0
700  34  0x10L   0x0L    1   0
740  36  0x14L   0x0L    1   0
780  38  0x4L    0x4L    1   1
820  40  0x4L    0x4L    1   1
860  42  0x4L    0x8L    1   1
900  44  0xcL    0x0L    1   0
940  46  0x10L   0x0L    1   0
980  48  0x14L   0x0L    1   0
```

```
<class 'myhdl._SuspendSimulation'>: Simulated 1000 timesteps
```

Test 3

In [1]:

```
from testbench import *
main()
```

```
t(ns)    cycle   output  addr    en  load
-----------------------------------------------
20   0   0x0L    0x0L    1   0
60   2   0x4L    0x0L    1   0
100  4   0x8L    0x0L    1   0
140  6   0xcL    0x0L    1   0
180  8   0x10L   0x0L    1   0
220  10  0x8L    0x8L    1   1
260  12  0x8L    0x8L    1   1
300  14  0xcL    0x0L    1   0
340  16  0x10L   0x0L    1   0
380  18  0x8L    0x8L    1   1
420  20  0xcL    0x0L    1   0
460  22  0x10L   0x0L    1   0
500  24  0x14L   0x0L    1   0
```

```
540  26  0xcL    0xcL    1   1
580  28  0x10L   0x0L    1   0
620  30  0x14L   0x0L    1   0
660  32  0x18L   0x0L    1   0
700  34  0x4L    0x4L    1   1
740  36  0x8L    0x0L    1   0
780  38  0x0L    0x0L    1   1
820  40  0x0L    0xcL    1   1
860  42  0x10L   0x0L    1   0
900  44  0x14L   0x0L    1   0
940  46  0x18L   0x0L    1   0
980  48  0x1cL   0x0L    1   0
```

```
<class 'myhdl._SuspendSimulation'>: Simulated 1000 timesteps
```

Test 4

In [1]:

```
from testbench import *
main()
```

```
t(ns)    cycle   output  addr    en  load
-----------------------------------------------
20   0   0x0L    0x0L    1   0
60   2   0x4L    0x0L    1   0
100  4   0x8L    0x0L    1   0
140  6   0xcL    0x0L    1   0
180  8   0x0L    0x0L    1   1
220  10  0x4L    0x0L    1   0
260  12  0x8L    0x8L    1   1
300  14  0xcL    0x0L    1   0
340  16  0x10L   0x0L    1   0
380  18  0x14L   0x0L    1   0
420  20  0x18L   0x0L    1   0
460  22  0x1cL   0x0L    1   0
500  24  0x20L   0x0L    1   0
540  26  0x0L    0x0L    1   1
580  28  0x0L    0x4L    1   1
620  30  0x8L    0x0L    1   0
660  32  0xcL    0x0L    1   0
700  34  0x10L   0x0L    1   0
740  36  0x14L   0x0L    1   0
780  38  0x0L    0x0L    1   1
820  40  0x0L    0xcL    1   1
860  42  0x10L   0x0L    1   0
900  44  0x4L    0x4L    1   1
```

```
940  46  0x4L    0xcL    1   1
980  48  0x10L   0x0L    1   0
```

```
<class 'myhdl._SuspendSimulation'>: Simulated 1000 timesteps
```

Test 5

In [1]:

```
from testbench import *
main()
```

```
t(ns)    cycle   output  addr    en  load
-----------------------------------------------
20   0   0x0L    0x0L    1   0
60   2   0x4L    0x0L    1   0
100  4   0x8L    0x0L    1   0
140  6   0xcL    0x0L    1   0
180  8   0x10L   0x0L    1   0
220  10  0x8L    0x8L    1   1
260  12  0x8L    0x8L    1   1
300  14  0xcL    0x0L    1   0
340  16  0x10L   0x0L    1   0
380  18  0x14L   0x0L    1   0
420  20  0x8L    0x8L    1   1
460  22  0xcL    0x0L    1   0
500  24  0x10L   0x0L    1   0
540  26  0x14L   0x0L    1   0
580  28  0x0L    0x0L    1   1
620  30  0x0L    0x4L    1   1
660  32  0x4L    0x0L    1   1
700  34  0x4L    0x0L    1   0
740  36  0x8L    0x0L    1   0
780  38  0x8L    0x8L    1   1
820  40  0x8L    0x4L    1   1
860  42  0x4L    0xcL    1   1
900  44  0x10L   0x0L    1   0
940  46  0x14L   0x0L    1   0
980  48  0x8L    0x8L    1   1
```

```
<class 'myhdl._SuspendSimulation'>: Simulated 1000 timesteps
```

Results

Test	Increments when EN = 1	Increments by 4	Handles Jump
1	Pass	Pass	Pass

Test	Increments when EN = 1	Increments by 4	Handles Jump
2	Pass	Pass	Pass
3	Pass	Pass	Pass
4	Pass	Pass	Pass
5	Pass	Pass	Pass

Conclusion

In all tests, the desired functionality can be observed. Therefore, the Program Counter operates as intended by design.