lab 3实验报告

20302010043 苏佳迪

一、流水线的改动

为了支持多周期乘除法器，流水线做出以下调整：

1、支持乘多周期除法器

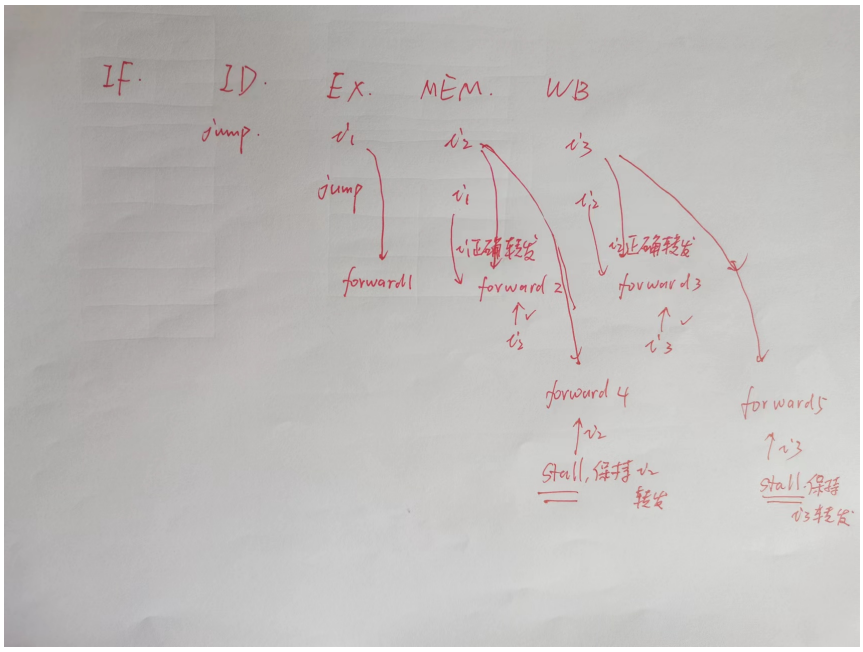
为了支持乘除法器的多周期运算，给`alu`和`execute`添加时钟与握手信号；当`execute_data_ok`为低电平时，说明`execute`在该周期未计算完成，此时需要阻塞`execute`以及之前的流水段，`memory`与`writeback`流水段继续流动并插入气泡，直到某个周期中`execute_data_ok`信号为高电平，恢复流动。

存在一种情况：`execute`计算乘除法，此时需要用到`memory`转发的数据，但`memory`为访存指令存在延迟，即`multiplier`和`divider`拿到的操作数据有可能是不对的，需要在拿到正确的数据后进行重新计算，解决方案为在满足上述条件时（具体到信号为`hazardOut.clear == 1`），将`multiplier`与`divider`的`reset`信号拉高，直到某一周期`memory`得到正确的结果`state_nxt`设置为`DOING`，下一周期开始计算。

2、转发器问题

`execute`阻塞，`memory`与`writeback`继续流动，此时会导致转发器数据覆盖，即memory会把writeback转发的数据覆盖掉（lab2时已经改进转发器解决无效数据转发的问题），若此时`execute`阻塞阶段需要用到writeback转发的数据就会导致执行错误。若通过`memory`阻塞的方式防止覆盖，在特定条件下会导致`ireq`和`dreq`循环访存一直阻塞，无法正常执行。

解决方案：添加两个备份转发器，当`execute`阻塞而`memory`不阻塞时，阻塞两个备份转发器，五个转发器进行转发；如下图：



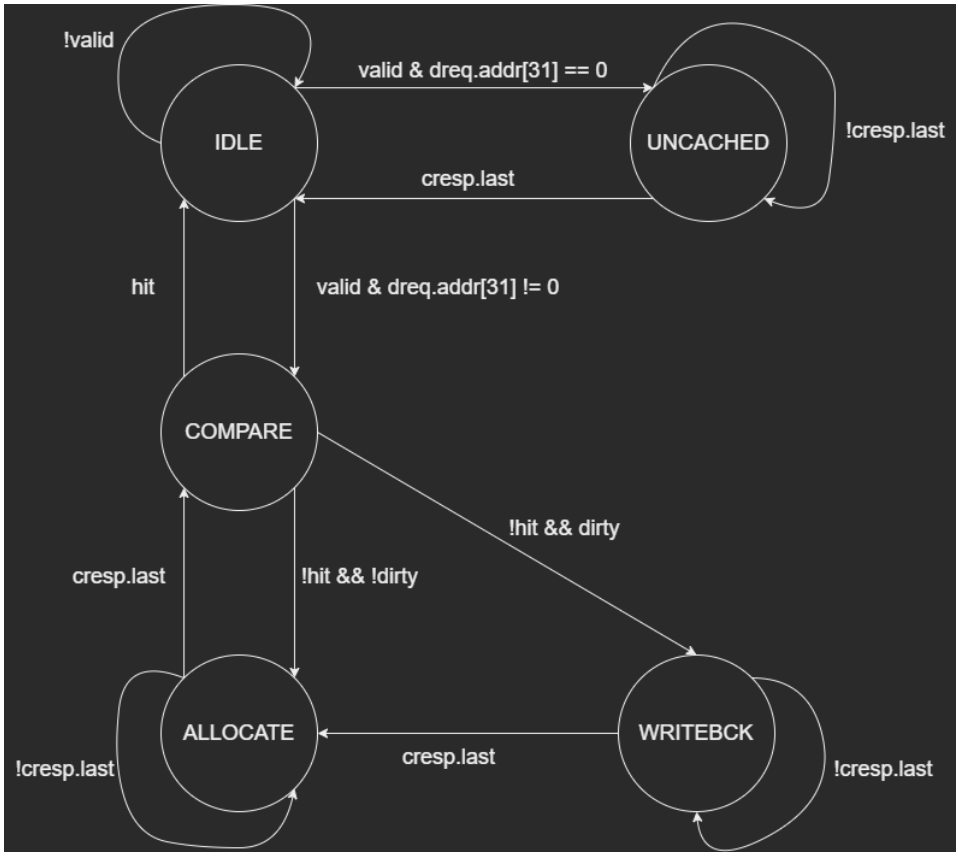
通过两个备份转发器，在一般情况下，`forward4`与`forward5`转发器与`forward2`和`forward3`转发器的数据同步，在特殊条件下阻塞：

- `execute`阻塞而`memory`不阻塞时：`forward4`与`forward5`阻塞，1~3正常运行，备份execute之前的几条指令的转发数据，防止数据覆盖导致的数据源丢失；
- `memory`阻塞时，`forward5`阻塞保持`i3`转发的数据，`forward3`正常转发接收`i2`转发的数据；

二、缓存的设计

1、状态机

缓存状态有五种：空闲`IDLE`、标志比较`COMPARE`,取数`FETCH`、写回`WRITEBACK`与不经过缓存`UNCACHED`，转化关系如下：



（hit状态下存在1个周期的延迟）

2、LRU替换策略

每个cache set定义一个组内数组`addr_t [ASSOCIATIVITY - 1 : 0] used_line`，数组大小为关联度`ASSOCIATIVITY`，每个元素下标代表组内编号为下标的块，**数据为该块的**`hit`**情况，数据范围从0到ASSOCIATIVITY - 1**，该信号只会在`COMPARE`状态并且hit条件下修改，修改逻辑如下：

- 如果`i == position`，那么把`used_line[i]`置为0，表示最近一次访问；
- 如果`used_line[i] >= used_line[position_nxt]`，那么`assign used_line[i] = used_line[i]`；
- 如果`used_line[i] < used_line[position_nxt]`，那么`assign used_line[i] = used_line[i] + 1`；

通过这样的逻辑来控制`used_line`的访问情况始终保持在最近访问为`0`，最远访问为`ASSOCIAVITY - 1`；每个cache set维护一个数组，用该数组表示最近访问的情况，数据越小表示越近访问，数据越大表示越远访问；当组内需要进行替换时，通过组合逻辑定位到max的块，确定该块为要替换的块，替换完成后状态回到COMPARE时，因为hit信号拉高，会维护组内的这个数组，按上述逻辑进行调整。

三、测试结果

1、verilator测试

```
[WARNING] difftest store queue overflow
Random test passed.
Single test passed.
Run paint
Aptenodytes patagonicus
Picture generated.
Compressed size=2162
Done! Result:

PgBiVkQAppeDAMu8qQDO4c4A9ebTAAYAAmsQuwjMAmIEIgJhIxECYBcAAAAGAAJrErsGzAJiBCICYSIRAmYYAAAABgACaxO7BcwCYgQiAmEiEQJgGAAAAAYAAmoHqg67A8wCZgQiAmEiEQJgGAAAAA
YAAmoLqgu7A8wCYgMiAmEhEQJgGQAAAAYAAmoNqgq7AswCYgMiAmEhEQJgGQAAAAYAAmoPqgm7AsYFIgJhHxECYBoAAAAHAAJqD6oJuwLGBCICYR8RAmAaAAAABwACahGqCLsCYgMiAmEeEQJmGwAA
AAcAAmkGmQyqB7sCYgMiAmEeEQJgGwAAAAcAAmkJmQqqB7sCYgMiAmEdEQJgGwAAAAcAAmkLmQmqBrsCYgMiAmEcEQJgHAAAAAcAAmkNmQiqBrsCYgMiAmEcEQJmHQAAAAcAAmkOmQiqBrsCYgIiAmYaEQJgHQAAAAcAAmgGiAmZB6oGuwJiAyICYRgRAmAeAAAABwACaAiICJkHqga7AmIDIgJhFxECYB4AAAAHAAJoCYgImQaqBrsCxgMiAmEWEQJmHwAAAAcA
AmgKiAeZB6oGuwJiAyICYRURAmAfAAAABwACaAuIB5kGqga7AsYEIgJhFBECYB8AAAAIAAJoC4gGmQaqBrsCzAJiAiICZhMRAmAgAAAACAACZwR3B4gGmQaqBrsDzAJiAiICZhIRAmAgAAAACAACZw
V3B4gGmQaqBbsEzAJiAiICZhARAmYhAAAACAACZw3BogGmQaqBrsEzAJiAiICZg8RAmAhAAAACAACZw3BogGmQaqBrsFzALWAmICIgJmCxECYCIAAAAHAAJmB3cGiAaZBqoGuwXMA90CZgIiAmYKEQJgIgAAAAgAAmcGdwaIBpkGqga7BcwE3QNmAgADZgYRAmAjAAAACAACZwZ3BogGmQaqBrsFzAXdAuYCZgIAA2YEEQJgIwAAAAgAAmcGdwaIBp
kGqga7BcwF3QPuA2YDAAVmJAAAAAgAAmYFdweIBpkGqgW7BswF3QXuAvYGZiUAAAAJAAJnA3cHiAaZBqoGuwbMBd0F7gT/BGYlAAAACQACaAqIBpkGqga7BswF3QXuA/8CZgJEAmAkAAAACQACaAmI
B5kGqga7BcwG3QXuAvYCZANEAmAkAAAACgACaAeIB5kHqga7BcwG3QTuAmYGRAJgJAAAAAoAAmgGiAiZBqoGuwbMBd0E7gJmB0QCYCQAAAAKAAJmBYgImQeqBrsGzAXdAuYCZglEAmAkAAAACwACaA
KICZkHqge7BcwG3QJkC0QCYCQAAAALAAJpCpkIqga7BswE3QJmDUQCYCQAAAAMAAJpCJkIqge7BswC3QJmAjYNRAJgJAAAAAwAAmkHmQiqB7sGzALdAmMCMwJkDEQCYCQAAAAMAAJmBpkJqge7BswC
ZgQzAmQMRAJgJAAAA0AAmkDmQqqB7sEzANmBjMCZAxEAmAkAAADQACZgyqB7sEzAJjCDMCZAxEAmAkAAADgACagqqCLsCzAJmCjMCZAxEAmAkAAADgACZgiqCbsCxgNmCjMCZgtEAmAlAAAADw
ACagaqCbsCxgJEAmMlMwJkCkQCYCUAAAAQAAJqA6oJuwNmAkQCZgwzAmQKRAJgJQAAABAAAmYCqwi7AmYERAJmDTMCZApEAmAlAAAAEQACawi7AmYFRAJjDTMCZApEAmAlAAAAEgACawW7AmYGRAJm
DjMCZApEAmAlAAAAEgACZgO7AmYIRAJjDjMCZApEAmAlAAAAEwACawK2AmQIRAJjDjMCZApEAmAlAAAAEwADZgpEAmYOMwJmC0QCYCUAAAAUAAJkCkQCYw4zAmQLRAJgJQAAABQAAmYKRANmDDMCZg
xEAmAlAAAAFQACZAxEBWYFMwNmDUQCYCUAAAAWAAJkEEQGZg9EAmAlAAAAFwACZCREAmAlAAAAGAACZCNEAmAlAAAAGAACZiJEAmYmAAAAGQACZiFEA2Y1AAAAGgACZiBEA2U1AAAAHAACZB5EA2UC
YCMAAAAdAAJmHUQCZQJWAmAiAAAAHgACZhxEAmUCVQJgIgAAACAAAmYbRAJlAlYCYCEAAAAhAANmGUQCZQJVAmAhAAAAIwADZhdEAmUCVQJmIQAAACUABGYURAJmA1UCYCAAAAAoAAVmEUQCZQJVAm
YgAAAALQAHZgtEAmUCVQJgHwAAADMAAmYKRAJmAlUCZh8AAAA0AAJmCkQCZQJVAmYeAAAANQADZghEAmYDVQJgHQAAADcAAmYIRAJmAlUCZh0AAAA4AANmB0QCZgJVAmAcAAAAOgACZghEA2UCYBsA
AAA7AAJmCEQDZhwAAAA8AANmB0QDZhsAAAA+AAJmCEQCZhoAAAA/AAJmCEQCYBkAAABAAANmBkQCZhkAAABCAAJmBkQCYBgAAABDAANmBEQCZhgAAABFAAJmBEQCYBcAAABGAAJmA0QCZhcAAABHAA
NmAkQCYBYAAABJAAJmAkYCYBUAAABKAAVmFQAAAEwAA2YVAAAABQADZgIGAmYCBgJmAwACYDgAA2YUAAAABQACYAIAAmAEAAJgAgACYDkAAmYUAAAABQADZgIGAmYCBgJmAgYCZlAAAAAHAAJgAgAK
YE8AAAAFAANmAgYCZgIGAmYFBk8AAABkAAAAZAAAAGQAAABkAAAAAE=
Run coremark
Running CoreMark for 10 iterations
2K performance run parameters for coremark.
CoreMark Size    : 666
Total time (ms)  : 689
Iterations       : 10
Compiler version : GCC9.4.0
seedcrc          : 0xe9f5
[0]crclist       : 0xe714
[0]crcmatrix     : 0x1fd7
[0]crcstate      : 0x8e3a
[0]crcfinal      : 0xfcaf
Finised in 689 ms.
==================================================
CoreMark Iterations/Sec 14
Run dhrystone
Dhrystone Benchmark, Version C, Version 2.2
Trying 10000 runs through Dhrystone.
Finished in 1619 ms
==================================================
Dhrystone PASS         10 Marks
                  vs. 100000 Marks (i7-7700K @ 4.20GHz)
Run stream
```

```
                  vs. 100000 Marks (i7-7700K @ 4.20GHz)
Run stream
-------------------------------------------------------------
STREAM version $Revision: 5.10 $
-------------------------------------------------------------
This system uses 8 bytes per array element.
-------------------------------------------------------------
Array size = 2048 (elements), Offset = 0 (elements)
Memory per array = 0.0 MiB (= 0.0 GiB).
Total memory required = 0.0 MiB (= 0.0 GiB).
Each kernel will be executed 10 times.
 The *best* time for each kernel (excluding the first iteration)
 will be used to compute the reported bandwidth.
-------------------------------------------------------------
* checktick: start=1.651739
* checktick: start=1.653423
* checktick: start=1.655070
* checktick: start=1.656827
* checktick: start=1.658511
* checktick: start=1.660224
* checktick: start=1.661934
* checktick: start=1.663589
* checktick: start=1.665365
* checktick: start=1.667163
* checktick: start=1.668996
* checktick: start=1.670813
* checktick: start=1.672593
* checktick: start=1.674368
* checktick: start=1.676092
* checktick: start=1.677791
* checktick: start=1.679516
* checktick: start=1.681260
* checktick: start=1.682961
* checktick: start=1.684699
Your clock granularity/precision appears to be 110 microseconds.
Each test below will take on the order of 16104 microseconds.
   (= 146 clock ticks)
Increase the size of the arrays if this shows that
you are not getting at least 20 clock ticks per test.
-------------------------------------------------------------
WARNING -- The above is only a rough guideline.
For best results, please be sure you know the
precision of your system timer.
-------------------------------------------------------------
Function    Best Rate MB/s  Avg time     Min time     Max time
Copy:               8.4     0.004226     0.003920     0.004416
Scale:              0.6     0.055487     0.055231     0.055731
Add:                2.1     0.024179     0.022979     0.025670
Triad:              0.7     0.072372     0.071304     0.074065
-------------------------------------------------------------
Solution Validates: avg error less than 1.000000e-13 on all three arrays
-------------------------------------------------------------
Run conwaygame
```

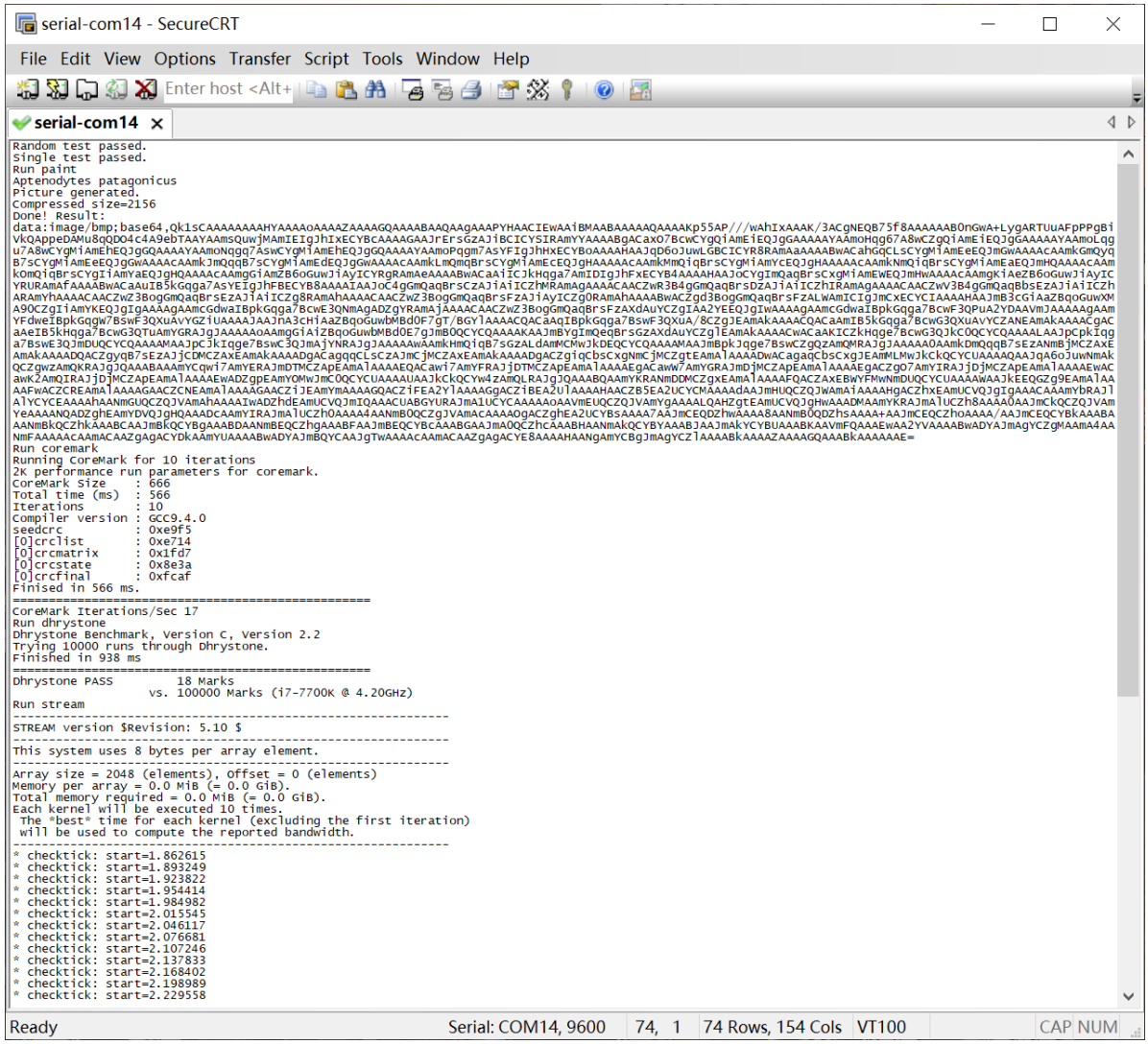
```
Solution Validates: avg error less than 1.000000e-13 on all three arrays
------------------------------------------------------------
Run conwaygame
Play Conway's life game for 200 rounds.
seed=3663


     *
    * *
    *  *        ***   ***
     **
              *    * *    *
              *    * *    *
              *    * *    *       **
                ***   ***         **
[src/cpu/cpu-exec.c,320,cpu_exec] nemu: HIT GOOD TRAP at pc = 0x0000000080014e44
[src/cpu/cpu-exec.c,321,cpu_exec] trap code:0
[src/cpu/cpu-exec.c,62,monitor_statistic] host time spent = 23361258 us
[src/cpu/cpu-exec.c,64,monitor_statistic] total guest instructions = 34605237
[src/cpu/cpu-exec.c,65,monitor_statistic] simulation frequency = 1481308 instr/s
sh: 1: spike-dasm: not found
```

在verilator仿真测试的环境下，`TEST=paint`用时2294，如下图：

`TEST=coremark`测试结果为：`Interations/Sec 14`；`TEST=dhrystone`测试结果为：`10 Marks`；`TEST=stream`的测试结果为`Copy Best Rate = 8.4MB/s`；`TEST=conwaygame`测试结果如上。

2、上板串口软件测试


serial-com14 - SecureCRT
File Edit View Options Transfer Script Tools Window Help
Enter host <Alt+
serial-com14
Random test passed.
Single test passed.
Run paint
Aptenodytes patagonicus
Picture generated.
Compressed size=2156
Done! Result:

Run coremark
Running CoreMark for 10 iterations
2K performance run parameters for coremark.
CoreMark Size : 666
Total time (ms) : 566
Iterations : 10
Compiler version : GCC9.4.0
seedcrc : 0xe9f5
[0]crclist : 0xe714
[0]crcmatrix : 0x1fd7
[0]crcstate : 0x8e3a
[0]crcfinal : 0xfcaf
Finised in 566 ms.
CoreMark Iterations/Sec 17
Run dhrystone
Dhrystone Benchmark, Version C, Version 2.2
Trying 10000 runs through Dhrystone.
Finished in 938 ms
Dhrystone PASS 18 Marks
vs. 100000 Marks (i7-7700K @ 4.20GHz)
Run stream
STREAM version $Revision: 5.10 $
This system uses 8 bytes per array element.
Array size = 2048 (elements), Offset = 0 (elements)
Memory per array = 0.0 MiB (= 0.0 GiB).
Total memory required = 0.0 MiB (= 0.0 GiB).
Each kernel will be executed 10 times.
The *best* time for each kernel (excluding the first iteration)
will be used to compute the reported bandwidth.
* checktick: start=1.862615
* checktick: start=1.893249
* checktick: start=1.923822
* checktick: start=1.954414
* checktick: start=1.984982
* checktick: start=2.015545
* checktick: start=2.046117
* checktick: start=2.076681
* checktick: start=2.107246
* checktick: start=2.137833
* checktick: start=2.168402
* checktick: start=2.198989
* checktick: start=2.229558
Ready
Serial: COM14, 9600
74, 1
74 Rows, 154 Cols
VT100
CAP NUM

serial-com14 - SecureCRT

File Edit View Options Transfer Script Tools Window Help

Enter host <Alt+

serial-com14

```
CoreMark Iterations/Sec 17
Run dhrystone
Dhrystone Benchmark, Version C, Version 2.2
Trying 10000 runs through Dhrystone.
Finished in 938 ms
==================================================
Dhrystone PASS          18 Marks
                  vs. 100000 Marks (i7-7700K @ 4.20GHz)
Run stream
------------------------------------------------------------
STREAM version $Revision: 5.10 $
------------------------------------------------------------
This system uses 8 bytes per array element.
------------------------------------------------------------
Array size = 2048 (elements), Offset = 0 (elements)
Memory per array = 0.0 MiB (= 0.0 GiB).
Total memory required = 0.0 MiB (= 0.0 GiB).
Each kernel will be executed 10 times.
 The *best* time for each kernel (excluding the first iteration)
 will be used to compute the reported bandwidth.
------------------------------------------------------------
* checktick: start=1.862615
* checktick: start=1.893249
* checktick: start=1.923822
* checktick: start=1.954414
* checktick: start=1.984982
* checktick: start=2.015545
* checktick: start=2.046117
* checktick: start=2.076681
* checktick: start=2.107246
* checktick: start=2.137833
* checktick: start=2.168402
* checktick: start=2.198989
* checktick: start=2.229558
* checktick: start=2.260145
* checktick: start=2.290724
* checktick: start=2.321287
* checktick: start=2.351874
* checktick: start=2.382433
* checktick: start=2.413020
* checktick: start=2.443603
Your clock granularity/precision appears to be 67 microseconds.
Each test below will take on the order of 12091 microseconds.
   (= 180 clock ticks)
Increase the size of the arrays if this shows that
you are not getting at least 20 clock ticks per test.
------------------------------------------------------------
WARNING -- The above is only a rough guideline.
For best results, please be sure you know the
precision of your system timer.
------------------------------------------------------------
Function    Best Rate MB/s  Avg time     Min time     Max time
Copy:            15.2     0.002161     0.002153     0.002167
Scale:            0.7     0.044408     0.044348     0.044528
Add:              2.8     0.017995     0.017397     0.019190
Triad:            0.9     0.057979     0.057325     0.059291
------------------------------------------------------------
Solution validates: avg error less than 1.000000e-13 on all three arrays
------------------------------------------------------------
Run conwaygame
Play Conway's life game for 200 rounds.
seed=5119
                    *
                   * *
                   * *
                    *




Exit with code = 0
```

Ready | Serial: COM14, 9600 | 74, 1 | 74 Rows, 154 Cols | VT100 | CAP NUM

在上板测试的环境下，`TEST=paint`用时1307，如下图：

`TEST=coremark`测试结果为：`Interations/Sec 17`；`TEST=dhrystone`测试结果为：`18 Marks`；`TEST=stream`的测试结果为`Copy Best Rate = 15.2MB/s`；`TEST=conwaygame`测试结果如上。