Appendix A
UCM: Counter Machine Program

This appendix contains the universal counter machine program used by Paul Chapman for his Life Universal Computer described in Chap. 3.

The following is the universal counter machine program used by Paul Chapman for his Life Universal Computer [1] described in Chap. 3. It is preset in the author's Counter Machine Simulator found at [2]. The comments on the left identify macros in Paul Chapman's symbolic source.

```
#C registers = 6          # = 2^1     * 3^1 * 5^0
#C opcodes = 12           # = 2^2     * 3^1 * 5^0
#C operands = 2           # = 2^1     * 3^0 * 5^0
#C passaddresses = 2      # = 2^(3-2) * 3^0 * 5^0
#C failaddresses = 8      # = 2^(5-2) * 3^0 * 5^0
#C base = 2               # prime number label
#                         # of first instruction
#C opcode
#C godel
#C exp
#C ret
#C a
#C b
#Start 00
00 DEC opcodes 01 03
01 INC godel 02
02 INC a 00
03 DEC a 04 05
04 INC opcodes 03
05 INC ret 06
06 INC ret 07
07 DEC base 08 13
08 DEC godel 09 10
09 INC b 07
```


P. Rendell, *Turing Machine Universality of the Game of Life*,
Emergence, Complexity and Computation 18, DOI 10.1007/978-3-319-19842-2

```
10 INC base 11
11 DEC b 12 16
12 INC godel 10
13 DEC b 14 15
14 INC base 13
15 INC a 07
16 DEC godel 17 24              #Iszero
17 INC godel 18
15 INC a 07
16 DEC godel 17 24              #Iszero
17 INC godel 18
18 DEC a 19 27 lt
19 DEC base 20 22
20 INC b 21
21 INC godel 19
22 DEC b 23 18
23 INC base 22
24 DEC a 25 26
25 INC godel 24
26 INC exp 07
27 NOP 28
28 DEC godel 28 29              #Clr:
29 DEC ret 30 81
30 DEC ret 31 39
31 DEC exp 32 33 :
32 INC opcode 31
33 DEC opcode 34 86
34 DEC operands 35 37
35 INC godel 36
36 INC a 34
37 DEC a 38 06
38 INC operands 37
39 INC exp 40
40 INC exp 41
41 DEC opcode 42 66
42 DEC exp 43 48
43 DEC registers 44 45
44 INC b 42
45 INC exp 46
46 DEC b 47 51
47 INC registers 45
48 DEC b 49 50
49 INC exp 48
50 INC a 42
51 DEC registers 52 59          #Iszero
```

```
52 INC registers 53
53 DEC a 54 61
54 DEC exp 55 57
55 INC b 56
56 INC registers 54
57 DEC b 58 53
58 INC exp 57
59 DEC a 60 74 :
60 INC registers 59
61 DEC failaddresses 62 64
62 INC godel 63
63 INC a 61
64 DEC a 65 79
65 INC failaddresses 64
66 DEC registers 67 68          #inInstruction
67 INC a 66
68 DEC a 69 74
69 DEC exp 70 72 70 INC b 71
71 INC registers 69
72 DEC b 73 68
73 INC exp 72
74 DEC passaddresses 75 77
75 INC godel 76
76 INC a 74
77 DEC a 78 79
78 INC passaddresses 77
79 NOP 80
80 DEC exp 80 07                #Branch
81 INC exp 82
82 INC exp 83
83 DEC base 83 84               #Clr:
84 DEC exp 85 00
85 INC base 84
86 HLT
```

Appendix B
UCM: Turing Machine Program

The following is the full listing of the program for a universal counter machine based on simulating a Turing machine as described in Chap. 9.

The following is the full listing of the program for a universal counter machine based on simulating a Turing machine as described in Chap. 9. It is preset in the author's Counter Machine Simulator found at [2].

```
# Example Turing machine. Started with the read/write
# head  over one of a number of '0'sbetween two '1's
# replaces in all the '0's between the '1's with '1's
# Transitions cycle: write, move, read, choose next
# transition.
# transition numbers are mapped to prime numbers but
# coded in nextIf0 and nextIf1 in an odd way,
# T2 coded 1, T3 coded 2 and T5 coded 3.
#   P   write   move  symDir  next:0  next:1
#T2    0(0)   R(2)      2      T2 (1)    T3 (2)
#T3    1(1)   L(0)      1      T3 (2)    T5 (3)
#T5    1(1)   L(0)      1      HLT(0)    HLT(0)
#C a
#C b
#C godel
#C base = 2
#C exp
#C symDir  = 60  # 2^(2) * 3^(1) * 5^(1)
#C nextIf0 = 18  # 2^(1) * 3^(2) * 5^(0)
#C nextIf1 = 108 # 2^(2) * 3^(3) * 5^(0)
#C ret = 0
#C tL = 2
#C tR = 2
Lrepeat DEC symDir 02 04        # godel := symDir
02      INC godel 03
```


P. Rendell, *Turing Machine Universality of the Game of Life*,
Emergence, Complexity and Computation 18, DOI 10.1007/978-3-319-19842-2

```
03      INC a Lrepeat
04      DEC a 05 Lgodel
05      INC symDir 04
        # godel = symDIR    : ret = 0
        # godel = nextstate : ret = 1
Lgodel      DEC base 07 12      #Godel Loop
07      DEC godel 08 09
08      INC b Lgodel
09      INC base 10
10      DEC b 11 15
11      INC godel 09
12      DEC b 13 14
13      INC base 12
14      INC a Lgodel            # a:= godel/base
15      DEC godel 16 23
16      INC godel 17
17      DEC a 18 26
18      DEC base 19 21
19      INC b 20
20      INC godel 18
21      DEC b 22 17
22      INC base 21
23      DEC a 24 25
24      INC godel 23
25      INC exp Lgodel
        # exp := power
26      NOP 27
27      DEC godel 27 28         #clear godel
28      DEC ret 29 d01          #Jump if symbol
29      DEC base 29 30          #clear base
30      DEC exp 31 halt         #0 = halt
31      DEC exp 32 36           #1->2
32      DEC exp 33 35           #2->3
33      INC base 34             #n ->n+2
34      INC base 35
35      INC base 36
36      INC base 37
37      INC base 38
38      DEC exp 37 Lrepeat
d01     DEC exp d02 d10
d02     INC a d03
        #
        # exp == symDir(base) dir*2+sym
        #      DECode a := symbol, b := direction
d03     DEC exp d04 d10
```

```
d04     DEC a d05 d10
d05     INC b d01
d10     DEC b cA1 dA1           # jump if move right
        # Move Left
dA1     DEC tL dA2 d20
dA2     INC exp dA1             # exp := tL
d20     DEC a d21 d22
d21     INC tL d22              # a written first time
d22     DEC exp d23 d24
d23     INC tL d21
        # tapeLeft now tapeLeft *2 + a
d24     DEC tR d25 d40
d25     INC a d26
d26     DEC tR d27 d40
d27     DEC a d28 d40
d28     INC exp d24
d40     DEC exp d41 g00
d41     INC tR d40              #tR := tR/2,
                                #a:= remainder
        # Move Right
cA1     DEC tR cA2 c20
cA2     INC exp cA1             # exp := tR
c20     DEC a c21 c22
c21     INC tR c22              # a written first time
c22     DEC exp c23 c24
c23     INC tR c21
        # tapeRight now tapeRight *2 + a
c24     DEC tL c25 c40
c25     INC a c26
c26     DEC tL c27 c40
c27     DEC a c28 c40
c28     INC exp c24
c40     DEC exp c41 g00
c41     INC tL c40              #tl := tl/2,
                                #a:= remainder
        #-
g00     DEC a g01 g11           # test symbol read
g01     DEC nextIf1 g02 g04
g02     INC a g03
g03     INC godel g01
g04     DEC a g05 g30
g05     INC nextIf1 g04
g11     DEC nextIf0 g12 g14
g12     INC a g13
g13     INC godel g11
```

```
g14     DEC a g15 g30
g15     INC nextIf0 g14
        #
        # godel = next instruction
g30     INC ret Lgodel
halt    HLT
        #- end -
```

Appendix C
Tag Productions

This appendix is the full listing of the tag productions for the 2-tag machine described in Chap. 3.

Below is the full listing of the tag productions described in Sect. 3.7.1.3 for Tag Machine version of Turing machine in Fig. 2.13. It took 3,128 production cycles to convert the initial word of eight letters each designated by four symbols into the final word of 88 letters.

The Initila word:

```
B11_, b11_, C11_, c11_, c11_, c11_, c11_, c11_
```

The final word:

```
B71_, E500, C71_, F500, c71_, f500, c71_, f500,
c71_, f500, c71_, f500, c71_, f500, c71_, f500,
c71_, f500, c71_, f500, c71_, f500, c71_, f500,
c71_, f500, c71_, f500, c71_, f500, c71_, f500,
c71_, f500, c71_, f500, c71_, f500, c71_, f500,
c71_, f500, c71_, f500, c71_, f500, c71_, f500,
c71_, f500, c71_, f500, c71_, f500, c71_, f500,
c71_, f500, c71_, f500, c71_, f500, c71_, f500,
c71_, f500, c71_, f500, c71_, f500, c71_, f500,
c71_, f500, c71_, f500, c71_, f500, c71_, f500
```

Productions:

```
B10_ → S10_
C10_ → D101, D100
S10_ → B51_, B50_
b10_ → s10_
c10_ → d101, d100, d101, d100
s10_ → b51_, b50_
```


P. Rendell, *Turing Machine Universality of the Game of Life*,
Emergence, Complexity and Computation 18, DOI 10.1007/978-3-319-19842-2

```
B11_ → D11_, d11_, d11_, d11_
C11_ → S11_
D11_ → B21_, E110
S11_ → C21_, F110
b11_ → d11_, d11_, d11_, d11_
c11_ → s11_
d11_ → b21_, e110
s11_ → c21_, f110
B20_ → S20_
C20_ → D201, D200, d201, d200
S20_ → B31_, B30_
b20_ → s20_
c20_ → d201, d200, d201, d200
s20_ → b31_, b30_
B21_ → D21_, d21_, d21_, d21_
C21_ → S21_
D21_ → B11_, E210
S21_ → C11_, F210
b21_ → d21_, d21_, d21_, d21_
c21_ → s21_
d21_ → b11_, e210
s21_ → c11_, f210
B30_ → D30_, d30_, d30_, d30_
C30_ → S30_
D30_ → B21_, E300
S30_ → C21_, F300
b30_ → d30_, d30_, d30_, d30_
c30_ → s30_
d30_ → b21_, e300
s30_ → c21_, f300
B31_ → S31_
C31_ → D311, D310, d311, d310
S31_ → B41_, B40_
b31_ → s31_
c31_ → d311, d310, d311, d310
s31_ → b41_, b40_
B40_ → S40_
C40_ → D401, D400, d401, d400
S40_ → B31_, B30_
b40_ → s40_
c40_ → d401, d400, d401, d400
s40_ → b31_, b30_
B41_ → S41_
C41_ → D411, D410, d411, d410
S41_ → B31_, B30_
```

```
b41_ → s41_
c41_ → d411, d410, d411, d410
s41_ → b31_, b30_
B50_ → D50_, d50_
C50_ → S50_
D50_ → B71_, E500
S50_ → C71_, F500
b50_ → d50_, d50_, d50_, d50_
c50_ → s50_
d50_ → b71_, e500
s50_ → c71_, f500
B51_ → S51_
C51_ → D511, D510
S51_ → B61_, B60_
b51_ → s51_
c51_ → d511, d510, d511, d510
s51_ → b61_, b60_
B60_ → S60_
C60_ → D601, D600
S60_ → B51_, B50_
b60_ → s60_
c60_ → d601, d600, d601, d600
s60_ → b51_, b50_
B61_ → S61_
C61_ → D611, D610, d611, d610
S61_ → B51_, B50_
b61_ → s61_
c61_ → d611, d610, d611, d610
s61_ → b51_, b50_
D100 → c50_, C50_, c50_
d100 → c50_, c50_
D101 → C51_, c51_
d101 → c51_, c51_
E110 → a20_, B20_, b20_
F110 → C20_, c20_
e110 → b20_, b20_
f110 → c20_, c20_
D200 → c30_, C30_, c30_
d200 → c30_, c30_
D201 → C31_, c31_
d201 → c31_, c31_
E210 → a10_, B10_, b10_
F210 → C10_, c10_
e210 → b10_, b10_
f210 → c10_, c10_
```

```
E300 → a20_, B20_, b20_
F300 → C20_, c20_
e300 → b20_, b20_
f300 → c20_, c20_
D310 → c40_, C40_, c40_
d310 → c40_, c40_
D311 → C41_, c41_
d311 → c41_, c41_
D400 → c30_, C30_, c30_
d400 → c30_, c30_
D401 → C31_, c31_
d401 → c31_, c31_
D410 → c30_, C30_, c30_
d410 → c30_, c30_
D411 → C31_, c31_
d411 → c31_, c31_
E500 → a70_, B70_, b70_
F500 → C70_, c70_
e500 → b70_, b70_
f500 → c70_, c70_
D510 → c60_, C60_, c60_
d510 → c60_, c60_
D511 → C61_, c61_
d511 → c61_, c61_
D600 → c50_, C50_, c50_
d600 → c50_, c50_
D601 → C51_, c51_
d601 → c51_, c51_
D610 → c50_, C50_, c50_
d610 → c50_, c50_
D611 → C51_, c51_
d611 → c51_, c51_
```

References

1. Chapman, P.: Life Universal Computer. http://www.igblan.free-online.co.uk/igblan/ca/ (2002)
2. Rendell, P.: Java Applet Counter Machine Simulator/Gol Counter Machine Generator. http://www.rendell-attic.org/gol/UCM/index.htm (2011)

Index

P. Rendell, *Turing Machine Universality of the Game of Life*,
Emergence, Complexity and Computation 18, DOI 10.1007/978-3-319-19842-2