Computer Architecture Recitation 1

BLG322E

14.03.2024

Question 1

We design a pipeline P to perform the task **T** on the elements of an array. Using this pipeline P, it takes

- $\mathbf{T}_2 = 150$ ns to execute the task on an array with **two** elements.
- $\mathbf{T}_3 = 180$ ns to execute the task on an array with **three** elements

Question 1

a) Calculate the number of stages of the pipeline P (k=?).

b) Without a pipeline, it takes $t_n = 90$ ns to execute the task on a single element. Given this information, calculate the speedup achieved by the pipeline P if the array has an infinite number of elements ($S_{n\to\infty}$ =?).

c) We design a new pipeline P_{new} for the same task **T**. P_{new} has more stages than the previous pipeline P given above. Using this new pipeline P_{new} , it takes $\mathbf{T}_{\text{new1}} = 120$ ns to execute the task on only the first element. Compare the speedups the two pipelines (P and P_{new}) can achieve for an array with an infinite number of elements. ($\mathrm{S}_{\text{new }\infty} > \mathrm{S}_{\infty}, \mathrm{S}_{\text{new }\infty} < \mathrm{S}_{\infty}$, or $\mathrm{S}_{\text{new }\infty} = \mathrm{S}_{\infty}$)?

Solution 1a

- $T_2 = 150$ ns
- $T_3 = 180$ ns
- $T_3 - T_2 = \mathrm{t_p} = \mathbf{30}\mathrm{ns}$ Cycle time (period of clock signal)
- $T_2 = (\mathrm{k}+1)\mathrm{t_p} \Rightarrow \mathbf{k} = \mathbf{4}$ Number of stages

Solution 1b

- $S_{n\to\infty} = \frac{t_n}{t_p} = \frac{90}{30}$
- $\boldsymbol{S_{n\to\infty} = 3}$

Solution 1c

- $T_1 = kt_p = 120ns$
- $T_{new1} = k_{new} \cdot t_{pnew} = 120ns$
- $k_{\text{new}} \;\; > k \quad \Rightarrow t_{pnew} < t_p$
- The cycle time of the new pipeline is shorter.
- $S_{\text{new}\infty} = \frac{t_n}{t_{pnew}}, \quad S_\infty = \frac{t_n}{t_p}$
- $t_{pnew} < t_p \quad \Rightarrow S_{\text{new}\infty} > S_\infty$

Question 2

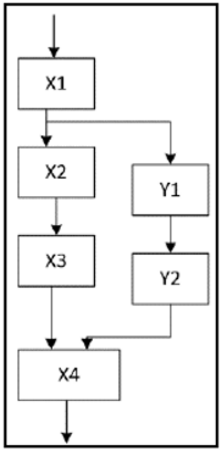
A task T that will be executed on integers consists of six suboperations X_i (i=1,2,3,4) and Y_j (j=1,2).

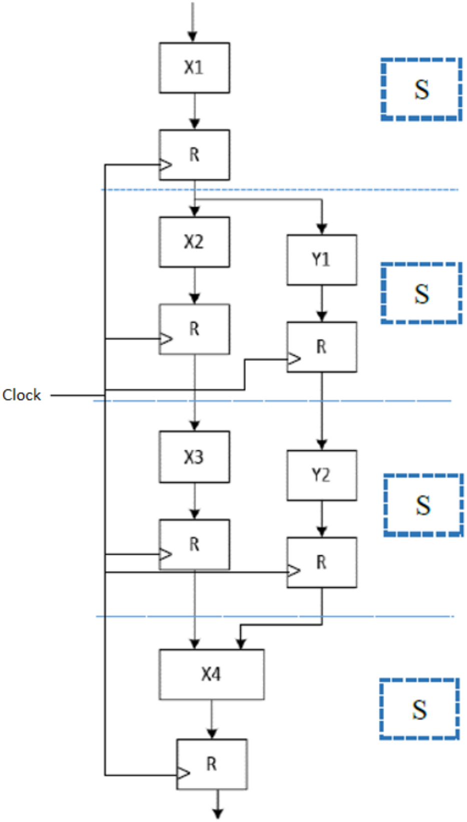
- These suboperations are implemented using combinational digital circuits with the following propagation delays: $X_1 = 25$ ns, $X_2 = 15$ ns, $X_3 = 20$ ns, $X_4 = 40$ ns, $Y_1 = 10$ ns, $Y_2 = 10$ ns.
- The block diagram of the circuit is shown on the next slide (left).
- The suboperations should be executed in the given order.
- Suboperation Y_1 can be executed in parallel with X_2 or X_3.
- Similarly, suboperation Y_2 can also be executed in parallel with X_2 or X_3.

To execute task T faster on elements of an array, we construct a pipeline with four stages (S_1, S_2, S_3, S_4) shown on the next slide (right).

- The pipeline has a register with a delay of 5 ns after each stage.

Question 2





Question 2

a) How long does it take to execute the task only on the first element using the pipeline (T_1) ?

b) What is the duration of the one task without the pipeline (t_n) ?

c) What is the fewest number of elements the array should have to achieve any speedup with this pipeline?

d) What is the speedup this pipeline achieves when it executes a task T on an array with an infinite number of elements?

e) Provide an alternative pipeline design that executes task **T**. This new pipeline should

- not yield a lower speedup than the current pipeline on an array with an infinite number of elements,
- use the fewest possible number of registers, and
- use the units in the original circuit.

f) How long does it take to execute the task only on the first element using the pipeline designed in (e) (T_1) ?

g) When executing task T on an array having an infinite number of elements, what speedup does the pipeline designed in (e) achieve?

Solution 2a

The pipeline has 4 stages (k = 4), and the duration of the slowest stage (t_p) is:

$$\mathrm{S1} = \mathrm{X1} + \mathrm{R} = 25 + 5 = 30 \text{ ns}$$

$$\mathrm{S2} = \mathrm{X2} + \mathrm{R} = 15 + 5 = 20 \text{ ns}$$

$$\mathrm{S3} = \mathrm{X3} + \mathrm{R} = 20 + 5 = 25 \text{ ns}$$

$$\mathrm{S4} = \mathrm{X4} + \mathrm{R} = 40 + 5 = 45 \text{ ns}$$

$$\Rightarrow \quad \mathrm{t}_p = \max\left(S_i\right) = 45 \text{ ns}$$

$$\mathrm{T}_1 = k \cdot \mathrm{t}_p = 4 \times 45 = \mathbf{180}\mathrm{ns}$$

Solution 2b

To calculate the time to execute the one task without the pipeline (t_n), we consider the longest path (without the registers).

$$\mathrm{t}_n = 25 + 15 + 20 + 40 = \boldsymbol{100ns}$$

Solution 2c

To achieve any speedup with the pipeline, the execution time without the pipeline should be larger than the execution time with the pipeline.

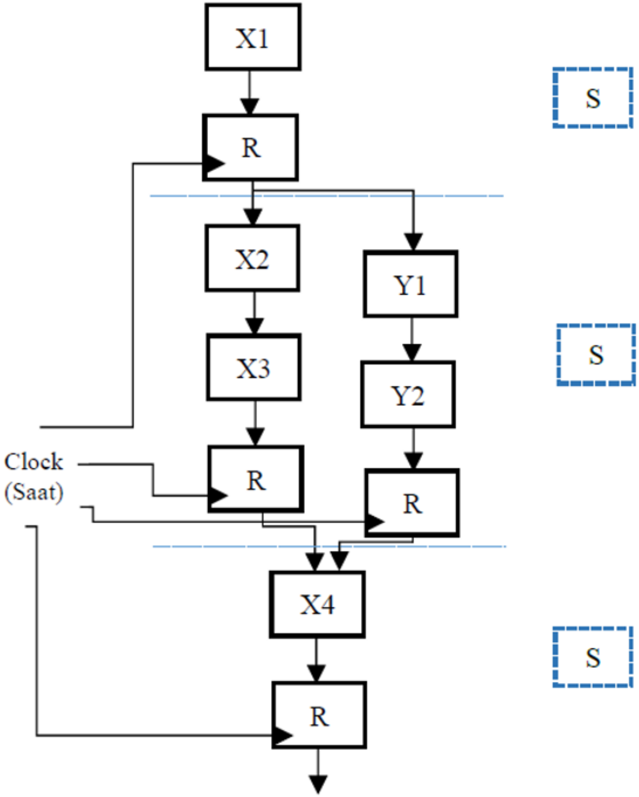
$$\begin{aligned} n \cdot \mathrm{t}_n &> (k + n - 1)\mathrm{t}_p \\ n \cdot 100 &> (4 + n - 1) \cdot 45 \\ n \cdot 100 &> 45n + 135 \\ 55n &> 135 \\ n &> 2.4545 \end{aligned}$$

Thus, the array should have at least 3 elements to achieve any speedup.

Solution 2d

$$\lim_{n \to \infty} S = \frac{\mathrm{t}_n}{\mathrm{t}_p} = \frac{100}{45} \approx 2.222$$

Solution 2e



Solution 2f

The pipeline has 3 stages ($k = 3$), and the duration of the slowest stage (t_p) is:

$$\begin{aligned} &\mathrm{S1} = \mathrm{X1} + \mathrm{R} = 25 + 5 = 30 \text{ ns} \\ &\mathrm{S2} = \mathrm{X2} + \mathrm{X3} + \mathrm{R} = 15 + 20 + 5 = 40 \text{ ns} \\ &\mathrm{S3} = \mathrm{X4} + \mathrm{R} = 40 + 5 = 45 \text{ ns} \\ &\Rightarrow \quad \mathrm{t}_p = \max\left(S_i\right) = 45 \text{ ns} \\ &\mathrm{T}_1 = k \cdot \mathrm{t}_p = 3 \times 45 = \mathbf{135}\text{ns} \end{aligned}$$

Solution 2g

$$\lim_{n\to\infty} S = \frac{\mathrm{t}_n}{\mathrm{t}_p} = \frac{100}{45} \approx 2.222$$

Question 3

You will design a pipeline that will execute the operation $[2(-A_i)]^3$, where A is an array that consists of 8-bit signed integers in two's complement. (For simplicity, assume that 32 bits are sufficient to represent all operation results.). You are allowed to use only the components that are given below with their timing attributes. You may access 2 different memory addresses simultaneously. You may use more than one of each unit if necessary.

Memory	(access time:	45 ns)
NOT gate	(propagation delay:	10 ns)
Adder	(propagation delay:	15 ns)
Shifter (combinatorial)	(propagation delay:	10 ns)
Multiplication circuit	(propagation delay:	45 ns)
Register	(delay:	5 ns)

Question 3

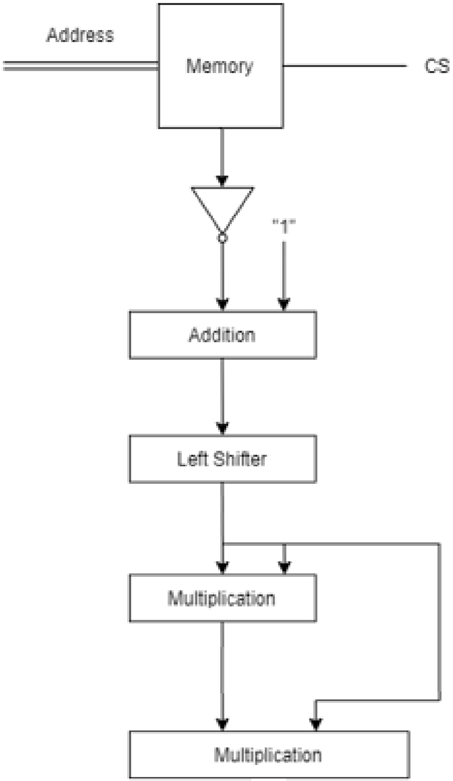
a) First, design and draw the circuit without a pipeline. Based on this design, calculate the duration of the operation without a pipeline. (Note: If a pipeline is not used, the required time of a task is the total delay of the combinatorial circuit elements on the longest path (without registers) in the designed structure.)

b) Now, design and draw the optimum pipeline that yields the highest speedup, while keeping the implementation cost as low as possible. Be sure take the waiting time for the first result into consideration. (Hint: Independent operations within the same stage can be performed in parallel.)

c) For the given propagation delay and access time information, calculate the speedup achieved for an array of 8 signed integers.

d) Calculate the theoretical maximum speedup of the pipeline you designed.

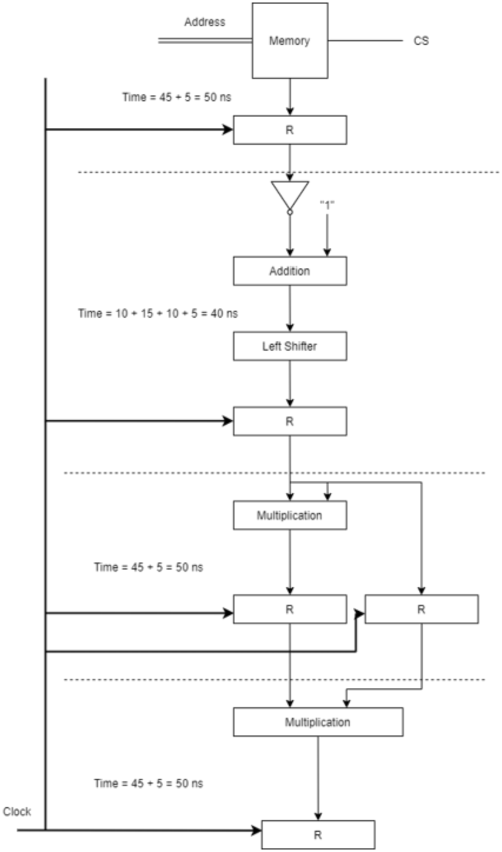
Solution 3a



Execution time without pipeline:

$t_n = 45 + 10 + 15 + 10 + 45 + 45 = 170$ ns

Solution 3b


Address
Memory
CS
Time = 45 + 5 = 50 ns
R
"1"
Addition
Time = 10 + 15 + 10 + 5 = 40 ns
Left Shifter
R
Multiplication
Time = 45 + 5 = 50 ns
R
R
Multiplication
Time = 45 + 5 = 50 ns
Clock
R

Solution 3c

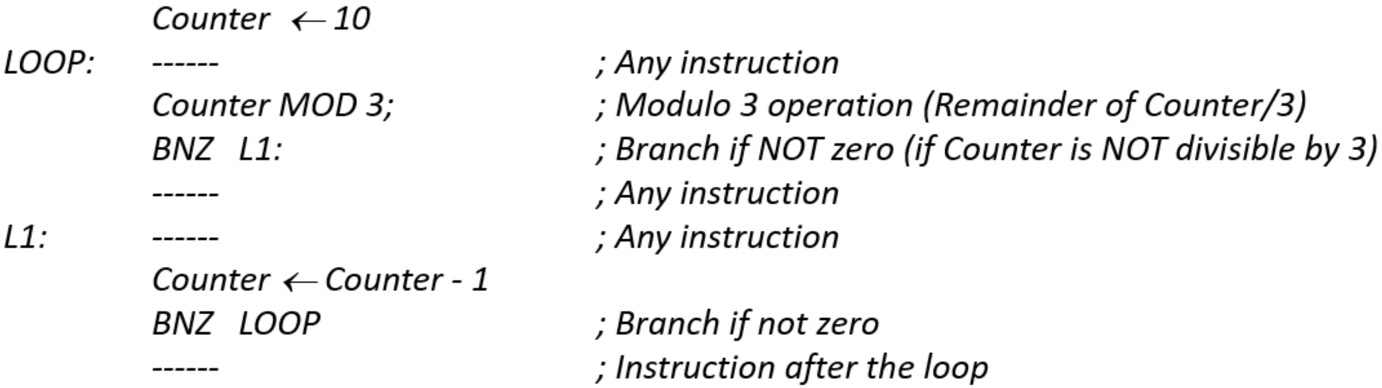
- Required time for the slowest stage is 50 ns. Thus, the cycle time should be the same (50 ns).
- Execution time without pipeline:
  $t_n = 45 + 10 + 15 + 10 + 45 + 45 = 170$ ns
- Execution time with pipeline:
  $(k + n - 1) \cdot t_p = (4 + 8 - 1) \times 50 = 550$ ns
- Speedup $= \frac{n \times t_n}{(k+n-1) \cdot t_p} = \frac{8 \times 170}{550} \approx 2.47$

Solution 3d

Theoretical maximum speedup=k=4

Question 4

A CPU that has an instruction pipeline with branch prediction mechanisms runs the piece of code given below:



```
        Counter ← 10
LOOP:   ------                 ; Any instruction
        Counter MOD 3;         ; Modulo 3 operation (Remainder of Counter/3)
        BNZ  L1:               ; Branch if NOT zero (if Counter is NOT divisible by 3)
        ------                 ; Any instruction
L1:     ------                 ; Any instruction
        Counter ← Counter - 1
        BNZ  LOOP              ; Branch if not zero
        ------                 ; Instruction after the loop
```

Fill in the tables below for both branch instructions that show

- the decision of the given prediction method,
- whether the branch is really taken or not, and
- if the prediction is correct or not

for each run of the instructions. Assume that the branch history table includes the branch target address in the beginning.

Question 4a

Dynamic prediction with one bit, initial decision is to take the branch

i) BNZ L1

#run	1	2	3	4	5	6	7	8	9	10
Counter										
Prediction (**T**aken or **N**ot)	T									
Really **T**aken or **N**ot										
Correct or **F**alse										

ii) BNZ LOOP

#run	1	2	3	4	5	6	7	8	9	10
Counter										
Prediction (**T**aken or **N**ot)	T									
Really **T**aken or **N**ot										
Correct or **F**alse										

```
        Counter ← 10
LOOP:   ------
        Counter MOD 3;
        BNZ  L1:
        ------
L1:     ------
        Counter ← Counter - 1
        BNZ  LOOP
        ------
```

Solution 4a

Dynamic prediction with one bit, initial decision is to take the branch

i) BNZ L1

#run	1	2	3	4	5	6	7	8	9	10
Counter	10	9	8	7	6	5	4	3	2	1
Prediction (**T**aken or **N**ot)	T	T	N	T	T	N	T	T	N	T
Really **T**aken or **N**ot	T	N	T	T	N	T	T	N	T	T
Correct or **F**alse	C	F	F	C	F	F	C	F	F	C

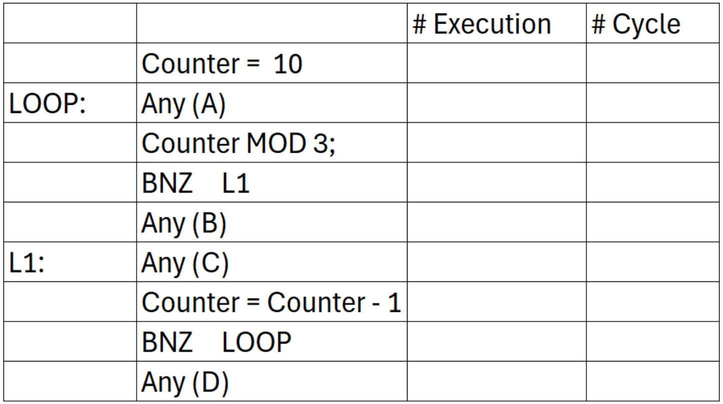
ii) BNZ LOOP

#run	1	2	3	4	5	6	7	8	9	10
Counter	9	8	7	6	5	4	3	2	1	0
Prediction (**T**aken or **N**ot)	T	T	T	T	T	T	T	T	T	T
Really **T**aken or **N**ot	T	T	T	T	T	T	T	T	T	N
Correct or **F**alse	C	C	C	C	C	C	C	C	C	F

```
        Counter ← 10
LOOP:   ------
        Counter MOD 3;
        BNZ   L1:
        ------
L1:     ------
        Counter ← Counter - 1
        BNZ   LOOP
        ------
```

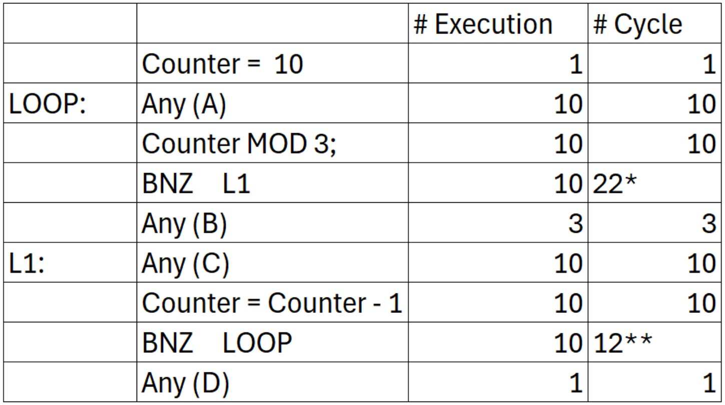
Question 4a (cont.)

Assume clock-cycles for any instruction is 1, but the branch misprediction penalty is 2 extra stall cycles. Evaluate CPI for the given program.



		# Execution	# Cycle
	Counter = 10		
LOOP:	Any (A)		
	Counter MOD 3;		
	BNZ L1		
	Any (B)		
L1:	Any (C)		
	Counter = Counter - 1		
	BNZ LOOP		
	Any (D)		

Solution 4a (cont.)



		# Execution	# Cycle
	Counter = 10	1	1
LOOP:	Any (A)	10	10
	Counter MOD 3;	10	10
	BNZ L1	10	22*
	Any (B)	3	3
L1:	Any (C)	10	10
	Counter = Counter - 1	10	10
	BNZ LOOP	10	12**
	Any (D)	1	1

- (*) For [BNZ L1]:
  #Cycle = 3 * #Misprediction + 1 * correct = 3 * 6 + 1 * 4 = 22
- (**) For [BNZ LOOP]:
  #Cycle = 3 * #Misprediction + 1 * correct = 3 * 1 + 1 * 9 = 12
- Total Cycle = 79 and Total Execution = 65
- CPI = 79 / 65 = 1.215

Question 4b

Dynamic prediction with two bits (saturating counter), initial decision is to take the branch (11)

i) BNZ L1

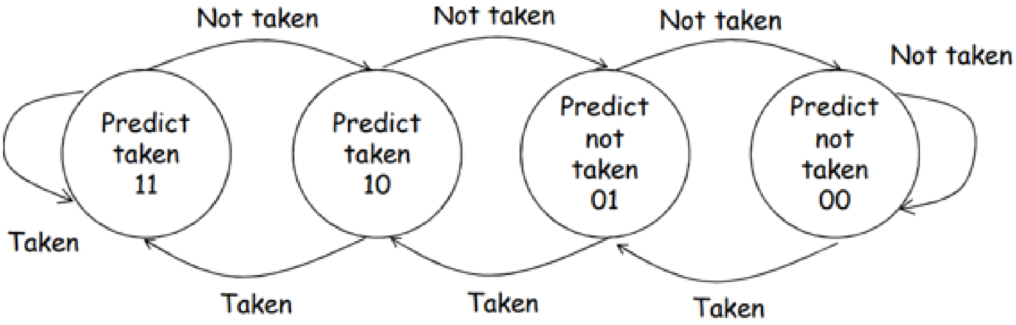
#run	1	2	3	4	5	6	7	8	9	10
Counter										
State	11									
Prediction (**T**aken or **N**ot)	T									
Really **T**aken or **N**ot										
Correct or **F**alse										

ii) BNZ LOOP

#run	1	2	3	4	5	6	7	8	9	10
Counter										
State	11									
Prediction (**T**aken or **N**ot)	T									
Really **T**aken or **N**ot										
Correct or **F**alse										

```
        Counter ← 10
LOOP:   ------
        Counter MOD 3;
        BNZ  L1:
        ------
L1:     ------
        Counter ← Counter - 1
        BNZ  LOOP
        ------
```

Question 4b


Not taken
Not taken
Not taken
Not taken
Predict taken 11
Predict taken 10
Predict not taken 01
Predict not taken 00
Taken
Taken
Taken
Taken

Solution 4b

Dynamic prediction with two bits (saturating counter), initial decision is to take the branch (11)

i) BNZ L1

#run	1	2	3	4	5	6	7	8	9	10
Counter	10	9	8	7	6	5	4	3	2	1
State	11	11	10	11	11	10	11	11	10	11
Prediction (**T**aken or **N**ot)	T	T	T	T	T	T	T	T	T	T
Really **T**aken or **N**ot	T	N	T	T	N	T	T	N	T	T
Correct or **F**alse	C	F	C	C	F	C	C	F	C	C

ii) BNZ LOOP

#run	1	2	3	4	5	6	7	8	9	10
Counter	9	8	7	6	5	4	3	2	1	0
State	11	11	11	11	11	11	11	11	11	11
Prediction (**T**aken or **N**ot)	T	T	T	T	T	T	T	T	T	T
Really **T**aken or **N**ot	T	T	T	T	T	T	T	T	T	N
Correct or **F**alse	C	C	C	C	C	C	C	C	C	F

```
        Counter ← 10
LOOP:   ------
        Counter MOD 3;
        BNZ   L1:
        ------
L1:     ------
        Counter ← Counter - 1
        BNZ   LOOP
        ------
```