Sakiko‘s Savings

Task

Sakiko was once the daughter of a wealthy family, but due to a family misfortune, her father became despondent and turned to alcohol. As a result, Sakiko had to start working to support the household. Each month, Sakiko earns a certain amount of money, while her father spends a certain amount. The specific rules are as follows:

1. Initially, Sakiko has savings of **10** units.
2. Initially, Sakiko earns **6** units per month.
3. Initially, her father spends **2** units per month.
4. Sakiko works very hard, so each month, Sakiko's earnings **double** from the previous month.
5. Each month, her father's spending **quadruples** from the previous month. However, if her father's spending **reaches or exceeds** Sakiko's earnings, Sakiko will admonish her father, causing his spending to reset to **2** units in the following month.
6. Calculate Sakiko's final savings after N months.

At the beginning of the i-th month, Sakiko's savings are denoted as $S(i)$. Sakiko's income for this month will be $Earn(i)$, and her father's expenditure for this month will be $Spend(i)$. The recursive formula is as follows:

$$S(n) = \begin{cases} 10, & \text{if } n = 0 \\ S(n-1) + Earn(n-1) - Spend(n-1), & \text{if } n > 0 \end{cases}$$

$$Earn(n) = \begin{cases} 5, & \text{if } n = 0 \\ Earn(n-1) \times 2, & \text{if } n > 0 \end{cases}$$

$$Spend(n) = \begin{cases} 2, & \text{if } n = 0 \\ 2, & \text{if } n > 0 \text{ and } Spend(n-1) \geq Earn(n-1) \\ Spend(n-1) \times 4, & \text{if } n > 0 \text{ and } Spend(n-1) < Earn(n-1) \end{cases}$$

Your Job

Note that N ($0 \leq N \leq 10$) will be stored in `x3100`.

You have to use **recursive** method to calculate Sakiko's final savings after N months, and store the result in the specified memory location `0x3200`.

R0-R7 are set to zeroes at the beginning, and your program should start at `x3000`.

For your convenience, your code may be written as:

```
.ORIG x3000

    LDI R0, INPUT

    ; Begin of your code
    ; ... ...
    ; ... ...
    ; End of your code

STACK   .FILL x6000
INPUT   .FILL X3100
RESULT  .FILL X3200

.END
```

You can modify this code and add it to the end of your code for testing.

```
.ORIG x3100
.FILL #5            ; Fill in x3100 with 5
.END
```

Score

Correctness for 50% and the report for other 50%.

Submission

Note that in this experiment, you are required to use **assembly code**.

Here are some notifications:

- Your program should start with `.ORIG x3000`.
- Your program should end with `.END`.
- Your last instruction should be `TRAP x25 (HALT)`.
- Include **comments** in your code where necessary for clarification.

Your submission should be structured as shown below:

```
PB********_Name.zip
├── PB********_Name_report.pdf
└── lab4.asm
```

Report

Your reports should contain at least the four parts below:

- purpose.
- principles.
- procedure (e.g. bugs you encountered and how to solve them).
- results of your test.
- answers to discussion questions.

Discussion Questions

- It is easy to see that this program can be rewritten as a simple iterative program, and the iterative method seems like more efficient compared to the basic recursive method. Why?
- Based on the reasons you mentioned, how can you improve the efficiency of this program(no code required)?