Open-source Software Lab

Project 1. TICO: Tiny Computer

13 Apr 2023

Shin Hong

Overview

- TICO is a simple computer with 12 instructions and 256 memory addresses
- Each team is asked to construct a C program that executes a given TICO assembly code (i.e., a TICO simulator)
  - programming requirements are given for you to exercise learnt C program features
- Also, you are asked to write TICO assembly programs to solve given problems, and show the results using your TICO simulators
- Submit the resulting source code and a video demo to HDLMS by 4 PM, 29 April

TICO: TIny COmputer

- memory is a map from addresses to values
- a memory address is a non-negative integer from 0 to 255
- a value is either a 1-byte integer number or instruction
  - integer number: -128 to 127
  - 15 instructions
    - receive input, produce output, read a value from a memory address, write a value to a memory address, arithmetic operation, jump, etc.
  - an uninitialized memory address has 0
  - unlike real computers, a value cannot be represented in 1 byte for TICO
- basically, TICO interprets instructions in sequence, except jump instructions
- first instruction must be placed at address 0
- for an invalid instruction, print the error message and terminate the execution.

Instruction	Description
READ [m]	receive a number from user and store it [m] (i.e., on address m)
WRITE [m]	print the number at [m]
ASSIGN [m] “c”	put a number c to [m]
MOVE [md] [ms]	put the value at [ms] to [md]
LOAD [md] [ms]	read an address stored at [ms], and put the value at the address to [md]
STORE [md] [ms]	read an address stored at [md], and put the value at [ms] to the address
ADD [md] [mx] [my]	add the values at [mx] and [my], and put the result to [md]
MINUS [md] [mx] [my]	subtract the value at [my] from the value at [mx], and put the result to [md]
MULT [md] [mx] [my]	multiply the value at [mx] by the value at [my] and put the result to [md]
MOD [md] [mx] [my]	Put the value at [mx] modulus the value at [my] to [md]
EQ [md] [mx] [my]	Put 1 to [md] if two values at [mx] and [my] are the same. Otherwise, put 0 to [md]
LESS [md] [mx] [my]	Put 1 to [md] if the value at [mx] is less than the value at [my]. Otherwise, put 0 to [md]
JUMP [m]	execute the instruction at [m] for the next turn
JUMPIF [m] [c]	execute the instruction at [m] for the next turn, if the value at [c] is not zero
TERM	terminate the program execution

TICO Assembly

- A TICO assembler program is a text file that contains a list of assembly instructions
- one text line corresponds to an instruction or value stored at a specific address
  - `<address> : <value>`
- Example: check whether a given number is even or odd

```
0: READ 21
1: EQ 23 21 22
2: JUMPIF 10 23
3: JUMPIF 7 24
4: MOVE 24 20
5: MINUS 21 21 20
6: JUMP 1
7: MOVE 24 22
8: MINUS 21 21 20
9: JUMPIF 1 20
10: WRITE 24
11: TERM
20: “1”
21:
22: “0”
23:
24: “1”
```

Another Example

```
0: READ 23
1: LESS 25 24 23
2: JUMPIF 4 25
3: MOVE 24 23
4: MINUS 20 20 21
5: LESS 23 22 20
6: JUMPIF 0 23
7: WRITE 24
8: TERM
20: “10”
21: “1”
22: “0”
23:
24: “127”
```

One More Example

```
0: MOVE 20 14
1: READ 21
2: STORE 20 21
3: ADD 20 20 13
4: LESS 22 20 15
5: JUMPIF 1 22
6: MINUS 20 20 13
7: LOAD 21 20
8: WRITE 21
9: LESS 22 20 14
10: JUMPIF 6 22
11: TERM
```

```
12: “0”
13: “1”
14: “30”
15: “40”
```

Programming Problems

1. Write a program that receives two positive integers, and then finds the greatest common divisor
2. Write a program that receives 8 binary numbers (0 or 1) and then convert them to the decimal number.
3. Write a program that receives 5 positive integers each of which is less than 20, and then prints out the round up of their average.
4. Write a program that receives 10 integer numbers, and then print out one of the numbers that appears most

TICO Simulator

- receive a filename of the assembler program as command-line argument
- receive a number from standard input for the READ instruction
- print out a number to standard output for the WRITE instruction
- print the error message and terminate for an invalid instruction
  - invalid memory address
  - arithmetic exceptions (e.g., overflow, underflow, divide-by-zero)

```
$ ./tico evenornot.txt
8
1
$
```

Simulator Workflow

- read an input file
  - convert each line to a value, and store it to the corresponding address
  - terminate the program if the input is not valid
- execute the program
  - set the cursor as 0
  - execute the instruction at the cursor, repeatedly
    - make an effect on the memory as the instruction commands
    - for JUMP and JUMPIF, update the cursor correspondingly
    - terminate the program if the instruction is TERM
    - if an instruction is invalid, print the error message and terminate
    - otherwise, increment the cursor by 1

Programming Requirements

- You must properly use `struct`, `enum`, `union` for representing “value”s
- You must use following string library functions appropriately:
  - `atoi`, `strtok`, `strcmp`
- You must write a Makefile for build script

Video Demo

- Demonstrate that your TICO simulator runs your solutions of the four problems successfully
- Describe the design and the implementation of your TICO simulator
- The demo video must take no more than 7 minutes.
- Each member must take a part in the video.
- Upload your demo video to YouTube, and write down the URL on the submission message

Evaluation

- Your result will be evaluated according to the following criteria:
  - all functionalities are correctly implemented
  - the source code is clean and comprehensible
  - all programming requirements are satisfied
  - all problems are solved with TICO assembly code
  - the demo video clearly describe that the program works correctly
- You will get extra points if you add new interesting instructions to TICO
- Good demo videos will be recognized

Other Instructions

- Open chat for Q&A - https://open.kakao.com/o/gII0Vjbf
- One submission for each team: no need to make redundant submissions.
- No late submission will be accepted.
- You must use English in recording a video demo
- Your demo video may be shared in the class, especially if it is to be recognized.