CSCI 6461 Web Assembler

User Guide for the Offline Two-Pass Browser Assembler

Use this guide with: CSCI6461_web_assembler.html

Purpose: The assembler translates CSCI 6461 assembly-language source into a listing file and a text load file containing an octal address and an octal 16-bit word on each generated line.

Document basis. This guide documents the current browser assembler and follows the August 2024 C6461 Instruction Set Architecture/Assembler specification.

Quick Start

1. Open CSCI6461_web_assembler.html in Firefox, Chrome, or Edge. No web server or Internet connection is required.
2. Click Open .asm to load an existing source file, or click Load Official Sample.
3. Edit the assembly source in the left pane.
4. Click Assemble, or press Ctrl+Enter while the source editor has focus.
5. Confirm that the status reports 0 errors.
6. Click Download Listing to save the .lst file.
7. Click Download Load File to save the octal address/word file for the simulator.

Critical number rule: Operands, LOC values, and DATA values are entered in decimal source notation. The listing and load-file address/word columns are displayed in octal.

1. Starting the Assembler

The assembler is a single HTML file. It runs completely inside the browser and does not install software or send source code to a server.

Opening the application

- Locate CSCI6461_web_assembler.html in File Explorer.
- Double-click it, or right-click and choose a browser.
- The official sample program is loaded and assembled automatically when the page opens.

Main screen

Area	Purpose
Toolbar	Opens, assembles, tests, saves, downloads, copies, or clears files.
Assembly Source	Editor for labels, directives, instructions, operands, and semicolon comments.
Assembler Status	Shows success, word and error counts, or line-numbered diagnostics.
Listing File	Shows source lines with octal address and generated machine word columns.
Load File	Shows only generated address/word pairs for loading into simulator memory.
Symbol Table	Shows each label, its six-digit octal address, and decimal address.

2. Toolbar Controls

Control	Action
Open .asm	Reads an .asm or plain-text source file. The program is assembled immediately after opening.
Load Official Sample	Replaces the editor contents with the official sample and assembles it.
Assemble	Performs both assembler passes and refreshes status, listing, load file, and symbol table.
Run Self-Test	Assembles the official sample internally and compares the generated load file with the expected 16-line result.
Save Source	Downloads the current editor contents as <program-name>.asm.

Control	Action
Download Listing	Downloads <program-name>.lst.
Download Load File	Downloads <program-name>.load as plain text.
Copy Load File	Copies the load-file text to the clipboard. The browser may block clipboard access for a local file.
Clear	Clears the source, outputs, status, and current filename.

Keyboard shortcuts: Tab inserts four spaces in the source editor. Ctrl+Enter (or Command+Enter on macOS) assembles the current source.

3. Writing Assembly Source

General line format

A source line can contain four fields. Only the operation field is required on an instruction line.

```
[Label:]  Operation  Operand[,Operand...]  ; Comment
```

Field	Form	Notes
Label	Start:	Begins with a letter or underscore, may contain letters, digits, or underscores, and ends with a colon.
Operation	LDR	Instruction mnemonic or the LOC/DATA assembler directive. Case is not significant.
Operands	0,1,12,1	Comma-separated decimal values or a label where permitted.
Comment	; explanation	Everything after the first semicolon is ignored by the assembler.

Labels and two-pass assembly

During pass 1, the assembler records label addresses. During pass 2, it generates machine words and resolves label references. A label may therefore be referenced before it is defined.

```
LOC 6
DATA End     ; forward reference is allowed
...
LOC 1024
End: HLT
```

Assembler directives

Directive	Example	Effect
LOC n	LOC 64	Sets the next memory location to decimal n. LOC does not generate a memory word.
DATA n	DATA 10	Generates one 16-bit word containing decimal n.
DATA label	DATA End	Generates one word containing the decimal address assigned to the label.

Memory range: LOC addresses must be decimal 0 through 2047. Each generated instruction or DATA line consumes one memory word.

Address fields: Most instructions have a five-bit address field, so the source address operand must resolve to decimal 0 through 31. Use an index register to reach higher memory addresses.

Signed DATA values

DATA accepts decimal values from -32768 through 65535. Negative values are stored as their 16-bit two's-complement representation.

4. Instruction Syntax Reference

Group	Mnemonics	Source form
Miscellaneous	HLT	HLT
Trap	TRAP	TRAP code
Memory	LDR STR LDA AMR SMR	op r,x,address[,I]
Index memory	LDX STX	op x,address[,I]
Conditional transfer	JZ JNE JCC SOB JGE	op r-or-cc,x,address[,I]
Unconditional/subroutine	JMA JSR	op x,address[,I]
Return	RFS	RFS immed
Immediate	AIR SIR	op r,immed
Register-to-register	MLT DVD TRR AND ORR	op rx,ry
Register unary	NOT	NOT rx
Shift/rotate	SRC RRC	op r,count,L/R,A/L
Input/output	IN OUT CHK	op r,devid

Group	Mnemonics	Source form
Floating/vector	FADD FSUB VADD VSUB CNVRT LDFR STFR	op r-or-fr,x,address[,I]

Optional indirect operand: The brackets in [,I] mean that the final indirect bit may be omitted. When omitted, I defaults to 0. When supplied, it must be 0 or 1.

Operand limits checked by the assembler

- General registers and condition-code selectors: 0 through 3.
- Index-register field: 0 through 3; LDX and STX require 1 through 3.
- Floating registers for FADD, FSUB, VADD, VSUB, LDFR, and STFR: 0 or 1.
- MLT and DVD rx and ry values: 0 or 2.
- Address and immediate fields: 0 through 31.
- TRAP code: 0 through 15; shift/rotate count: 0 through 15; device ID: 0 through 31.

5. Assembling a Program

1. Open the source file or enter the program in the Assembly Source pane.
2. Click Assemble.
3. Read the Assembler Status pane. Do not use the load file unless the error count is zero.
4. Review the listing. Every generated line should show a six-digit octal address and six-digit octal machine word.
5. Review the symbol table to confirm that labels were placed at the intended addresses.
6. Download the listing and load file.

Understanding the outputs

```
000016 102207  LDX 2,7      ; X2 GETS 3
```

In a listing line, the first column is the memory address in octal, the second column is the generated 16-bit word in octal, and the remainder is the original source line.

```
000016 102207
```

The load file contains only the two octal columns. It does not contain comments, blank lines, LOC directives, or other source text.

What happens when there is an error

The status pane lists each error by source line. The listing places ?????? in the generated-word column and appends an error comment. A line with an error is omitted from the load file.

```
000006 ??????  LDR 4,0,10  ; *** ERROR: register must be between 0 and 3
```

Do not run a partial load file: Although valid lines may still appear in the Load File pane, correct every reported error and assemble again before loading the simulator.

Common diagnostics

Message type	Likely correction
Unknown operation	Correct the mnemonic spelling or use a supported instruction.
Undefined label	Define the label, or correct the label spelling.

Duplicate label	Give each label a unique name.
Wrong operand count / syntax	Check commas and the instruction form in the reference table.
Value outside allowed range	Use the required register, address, immediate, count, code, or device range.
Memory address generated more than once	Correct LOC usage so two source lines do not occupy the same memory location.
Address exceeds installed memory	Keep generated locations within decimal 0 through 2047.

6. Saving Files

The application uses the opened source filename as the base name. For example, opening TestProgram.asm produces these downloads:

```
TestProgram.asm    source
TestProgram.lst    assembler listing
TestProgram.load   simulator load text
```

Browser downloads: The files normally appear in the browser's Downloads folder. If the browser asks for permission to download multiple files, allow it for this local assembler page.

7. Loading the Program into the Simulator

1. Open the current C6461 front-panel simulator HTML file.
2. Click Init.
3. Select the assembler-generated load file.
4. If the file chooser filters out .load files, choose All Files, or make a copy with a .txt extension. The .load file is plain text.
5. After the simulator reports that memory loading is complete, set the PC to the program's entry address in octal.
6. Use SS to test one instruction at a time or Run to execute continuously.

Source versus load file: Do not load the human-readable assembly source into simulator memory. Load the two-column octal file produced by Download Load File.

8. Official Sample and Self-Test

The official sample demonstrates LOC, DATA, a forward label reference, indexing, indirect addressing, and HLT.

```
; CSCI 6461 official assembler example
         LOC 6
         DATA 10
         DATA 3
         DATA  End
         DATA  0
         DATA  12
         DATA  9
         DATA  18
         DATA  12
         LDX 2,7
         LDR 3,0,10
         LDR 2,2,10
         LDR 1,2,10,1
         LDA 0,0,0
         LDX 1,8
         JZ 0,1,0
         LOC 1024
End:     HLT
```

Using the self-test

1. Click Run Self-Test.
2. The assembler internally assembles the official sample.
3. A passing result states that the expected 16-line octal load file was produced.
4. The self-test validates the assembler's sample translation; it does not test simulator instruction execution.

9. Recommended Development Workflow

Step	Practice
Write a small source block	Begin with a few instructions and comments.
Assemble	Correct all syntax, label, range, and location errors.
Review listing and symbols	Confirm addresses and machine words before execution.
Download source, listing, and load file	Keep the three files together under the same base name.
Load the simulator	Set the correct entry PC and single-step the first run.
Record expected results	For tests, document final PC, registers, CC/MFR, memory results, and terminal output.
Expand gradually	Add one tested feature or instruction group at a time.

10. Quick Checklist

☐ Source operands are decimal.
☐ Each label ends with a colon where it is defined.
☐ Comments begin with a semicolon.
☐ LOC and generated locations remain within decimal 0–2047.
☐ Five-bit instruction addresses and immediates resolve to decimal 0–31.
☐ Assembler status reports 0 errors.
☐ Listing addresses and words are six-digit octal values.
☐ The simulator receives the two-column load file, not the source file.
☐ The PC is set to the program entry address before execution.