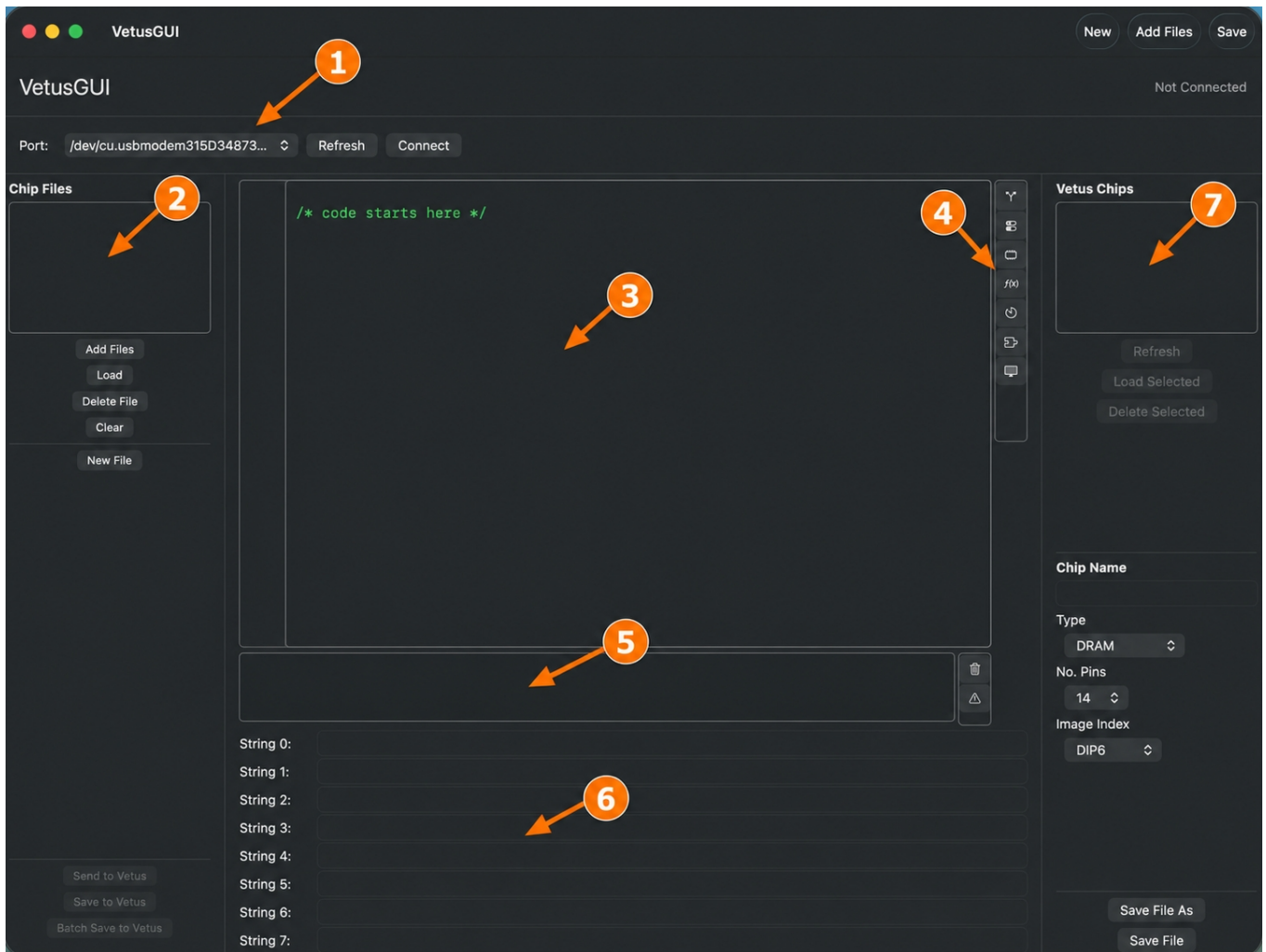


# VetusGUI User and Coding Manual

*A practical guide to connecting Vetus, creating chip files, writing tests, and managing images and fonts*



*The VetusGUI main window. Numbered areas are explained in Chapter 2.*

Version note: VetusGUI and Vetus firmware may continue to change. Items labeled “current behavior” describe the current implementation used for this manual.

# Contents

|    |                                    |
|----|------------------------------------|
| 1  | Getting Started                    |
| 2  | Main Window Tour                   |
| 3  | Chip File Workflow                 |
| 4  | Connecting and Working with Vetus  |
| 5  | Writing and Checking Test Code     |
| 6  | Code Builder and Opcode Guide      |
| 7  | Strings and Display Messages       |
| 8  | Image Manager                      |
| 9  | Font Manager                       |
| 10 | Settings                           |
| 11 | File Formats and Limits            |
| 12 | Troubleshooting and Safe Practices |
|    | Appendix Quick Reference           |

## 1 Getting Started

VetusGUI is the application used to prepare Vetus chip-test files, check and edit test code, keep a local library of chip files, and transfer chip tests, images, fonts, and settings to a connected Vetus unit. You can work on local chip files while disconnected. Device lists, transfers, and settings require a connection.

### Before you begin

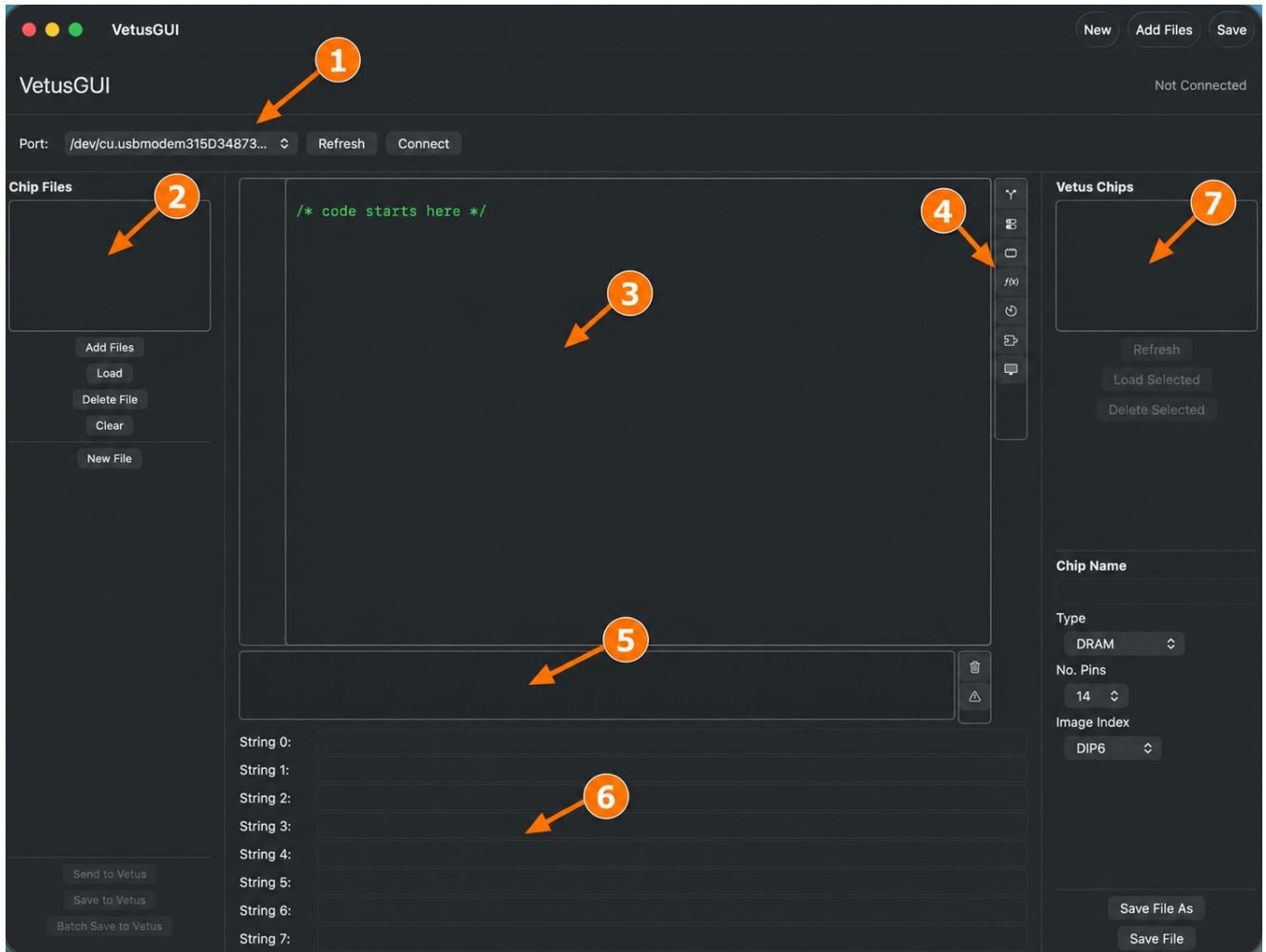
- A Mac, Windows PC or Linux PC running VetusGUI.
- A Vetus unit connected by a data-capable USB cable.
- A chip-test file to open, or enough information to create one: chip name, category, pin count, image, test code, and up to eight display strings.

### First connection

1. Connect Vetus to the computer and turn Vetus on.
2. Open VetusGUI.
3. Click Refresh beside the Port menu.
4. Choose the Vetus serial port. The name will depend on which OS you are using.
5. Click Connect. Confirm that the upper-right status changes to Connected and shows the Vetus serial number.

If no port appears, reconnect the USB cable, click Refresh, and verify that the cable carries data. Only transfer or settings controls that need Vetus are enabled after connection.

## 2 Main Window Tour



| No. | Area                               | What it is for                                                                                           |
|-----|------------------------------------|----------------------------------------------------------------------------------------------------------|
| 1   | Connection bar                     | Choose a USB port, refresh the port list, and connect or disconnect.                                     |
| 2   | Chip Files                         | Your local working list of .chip files. Add, load, remove from the list, or start a new file.            |
| 3   | Code editor                        | Write and edit the test program. Line numbers and syntax colors help you read it.                        |
| 4   | Code-building tools                | Open Control Flow, GPIO, Memory, Math, Timing, Add-on Boards, or Display helpers.                        |
| 5   | Terminal                           | Shows connection, load, save, compiler, and device messages. Trash clears it; warning lists code errors. |
| 6   | Strings 0 through 7                | Eight reusable messages that SHOSTR can display on Vetus.                                                |
| 7   | Vetus content and chip information | Browse tests stored on Vetus and edit the current chip name, type, pins, and image.                      |

## Top toolbar and menus

| Control     | Action                                                                       |
|-------------|------------------------------------------------------------------------------|
| New         | Starts a blank chip document after offering to save unsaved changes.         |
| Add Files   | Adds one or more local chip files to the Chip Files list.                    |
| Save        | Saves to the current local file; a new document opens Save As.               |
| File menu   | Load Chip File Command O; Save Command S; Save As Command Shift S.           |
| Device menu | Refresh Vetus Chips Command R; Connect or Disconnect Command C.              |
| Tools menu  | Image Window Command Shift I; Font Window Command Shift F; Delete All Chips. |
| Settings    | Opens device preferences for logo, LEDs, AUX, and default font.              |

## 3 Chip File Workflow

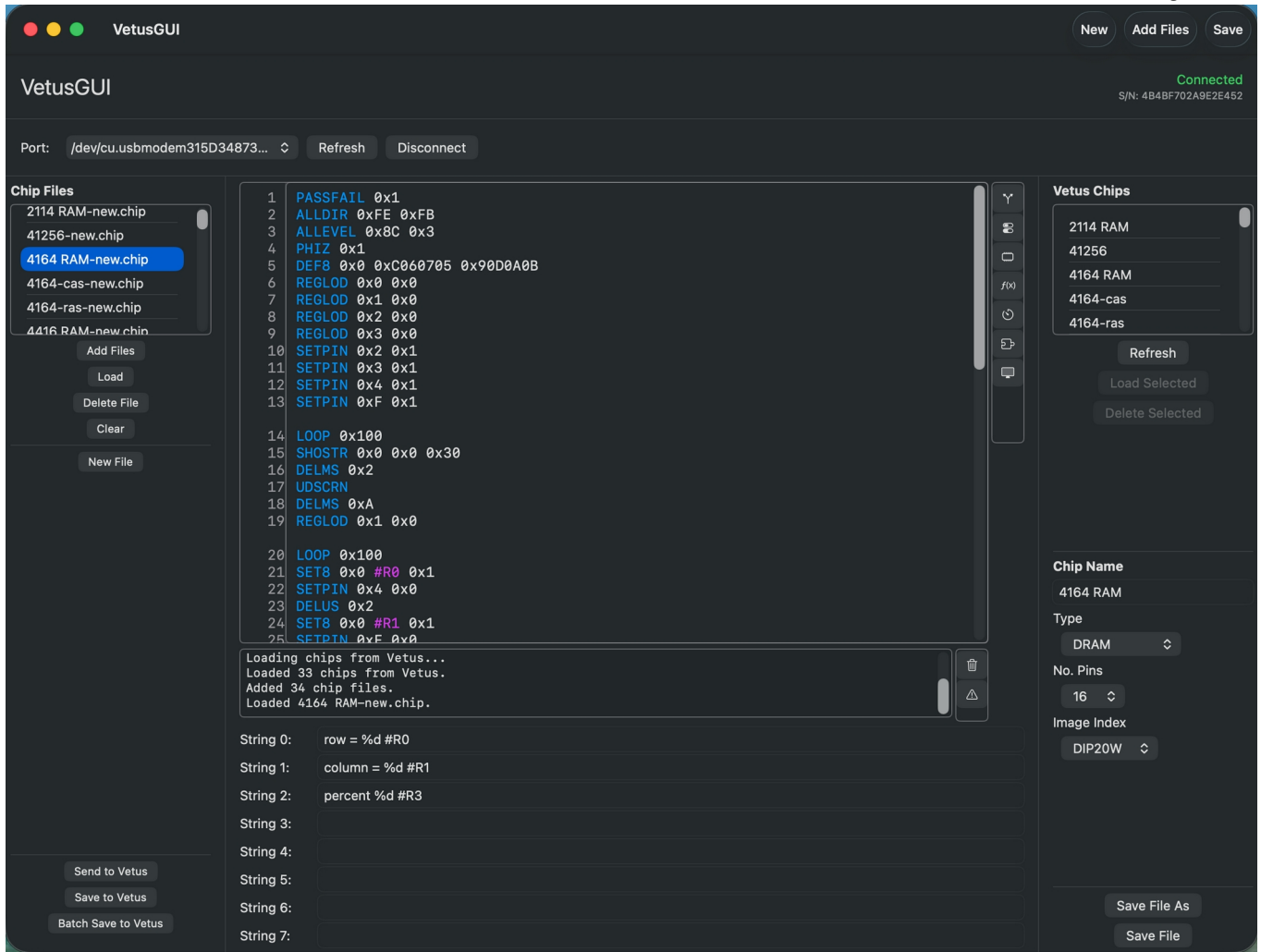
### Create a new chip file

1. Click New or New File. If the current document changed, choose Save, Save As, Don't Save, or Cancel.
2. Enter a Chip Name.
3. Choose a Type: DRAM, SRAM, CPU, 7400 Series, I/O, or Other. Unknown is not accepted when transferring.
4. Choose an even pin count from 2 through 40.
5. Choose the matching package picture under Image Index.
6. Write or insert test code and fill any String fields used by the code.
7. Click Save File As and choose a local filename. VetusGUI uses the .chip format.

### Open and manage local files

| Button       | Current behavior                                                                                         |
|--------------|----------------------------------------------------------------------------------------------------------|
| Add Files    | Selects one or more files and adds them to the left list. It does not automatically load their contents. |
| Load         | Loads the selected file into the editor, strings, and chip-information fields.                           |
| Delete File  | Removes the selected item from the working list. It does not delete the original file from disk.         |
| Clear        | Clears the local working list.                                                                           |
| Save File    | Writes the current document to its existing location; if none exists, Save As opens.                     |
| Save File As | Writes a new copy at a location you choose.                                                              |

Unsaved-change protection: New and file-loading actions compare the current content with the last clean copy. If it changed, VetusGUI asks whether to save before continuing.



A local 4164 file loaded in the editor with its strings and chip information.

## 4 Connecting and Working with Vetus

### Understand the two libraries

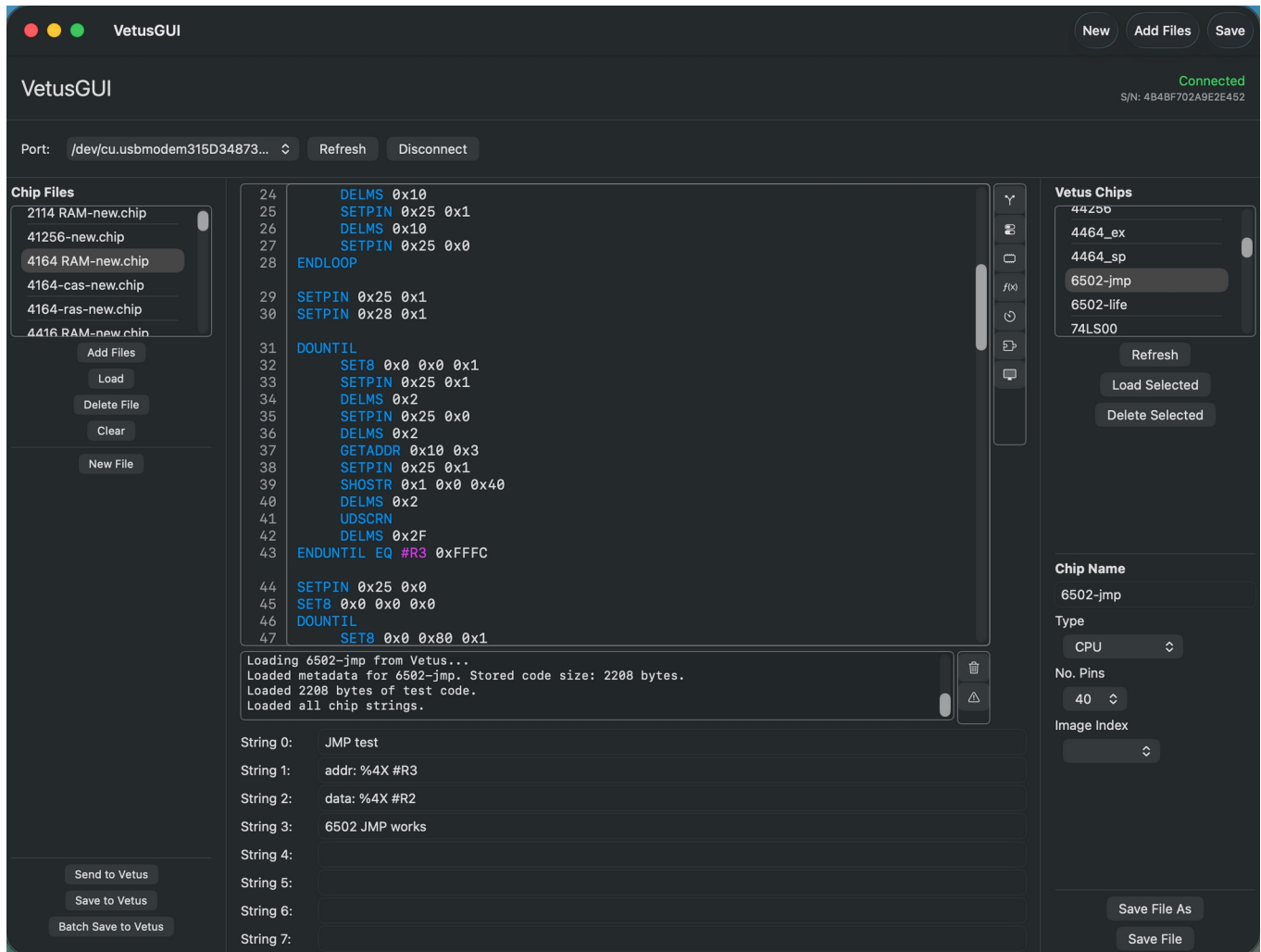
Chip Files on the left are local files available to this VetusGUI session. Vetus Chips on the right are tests stored in the connected unit. Loading from one side does not automatically save to the other.

### Transfer choices

| Action                    | Use it when                                                      | Result                                                                                                |
|---------------------------|------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------|
| Send to Vetus             | You want to run or use the current test temporarily.             | Compiles and sends the current chip document to Vetus without adding it to the stored chip list.      |
| Save to Vetus             | You want the current chip available from Vetus later.            | Compiles and writes the current chip document into Vetus storage.                                     |
| Batch Save to Vetus       | You want to store every valid file in the local Chip Files list. | Loads and saves the local documents as a batch. Watch the terminal for progress or a stopped item.    |
| Refresh under Vetus Chips | You want the latest stored list.                                 | Requests the chip list from Vetus.                                                                    |
| Load Selected             | You want to inspect or edit a stored test.                       | Loads metadata, compiled code, and all eight strings from Vetus, then shows decompiled/editable code. |

| Action           | Use it when                                | Result                                                                 |
|------------------|--------------------------------------------|------------------------------------------------------------------------|
| Delete Selected  | You want to remove one stored test.        | Deletes the selected Vetus entry after the device accepts the request. |
| Delete All Chips | You want to empty the device chip library. | Available from Tools and requires confirmation. This is destructive.   |

Before Send or Save, VetusGUI requires a chip name, a non-Unknown type, a pin count between 2 and 40, and code that compiles. Errors appear in the terminal.



A 6502 test loaded from Vetus. The terminal confirms metadata, code bytes, and strings were received.

## 5 Writing and Checking Test Code

Test code is written one statement per line. Opcode names are shown in blue, register references in purple, comments in green, and an unrecognized first word in red. Indentation and blank lines are kept in local files, which makes loops and functions easier to read.

### Basic syntax

- Opcode first, followed by arguments separated by spaces. Example: `SETPIN 0x4 0x1`.
- Numbers may be decimal, hexadecimal with `0x`, or binary with `0b` where accepted.
- Registers are R0 through R31. In value positions, reference their contents as `#R0` through `#R31`.
- DIM creates a named constant or value that can be used in later statements.
- Block comments use `/*` and `*/`. Comments are not executed.

- Condition operators accepted by the builder are EQ, NE, LT, GT, LE, GE, and DIV2. DIV2 is labeled as the even-number test in the picker.

Current behavior: when a file or device test is loaded, numeric condition values used by IF, WHILE, and ENDUNTIL are displayed as the matching named operator when possible. Re-check the terminal after loading older code.

## Named values and constants

```
DIM ROWS 0x100
DIM WRITE 0x1
REGL0D 0x0 0x0
LOOP ROWS
    SET8 0x0 #R0 WRITE
    REGINC 0x0
ENDLOOP
```

The editor supports named DIM values and register references in argument fields. This is the operator/constant support currently present in VetusGUI; the terminal remains the final check for whether a particular firmware build accepts the compiled result.

## Check code before transfer

- Click the warning triangle beside the terminal.
- Read each line-specific error.
- Correct red opcode words, missing arguments, unmatched blocks, invalid functions, or invalid values.
- Run the check again until no errors are listed.
- Send or save only after the terminal reports success.

The trash icon clears old terminal messages. Clearing the terminal does not change code or device content.

# 6 Code Builder and Opcode Guide

## Insert code without memorizing syntax

- Click one of the seven icons beside the editor. Hover over an icon to see its category name.
- Choose the action by its plain-language description.
- Fill the arguments. Chip-package helpers let you click pins for pin-based commands. Register menus cover R0 through R31.
- For a condition, choose an operator from the menu.
- Click Insert. The statement is placed at the current editor selection. Use Back to select a different action or Close to dismiss the panel.

Tools hides or shows the category strip. The categories are Control Flow, GPIO, Memory, Math, Timing, Add-on Boards, and Display.

## Control Flow

| Opcode  | Builder description | Main arguments |
|---------|---------------------|----------------|
| GOSTRT  | Go to code start    | None           |
| LOOP    | Start counted loop  | Loop count     |
| ENDLOOP | End counted loop    | None           |
| BRKLOOP | Break loop          | None           |

| Opcode   | Builder description   | Main arguments             |
|----------|-----------------------|----------------------------|
| WHILE    | Start while loop      | Operator, value 1, value 2 |
| ENDWHL   | End while loop        | None                       |
| JMP      | Jump to line number   | Line                       |
| DUNTIL   | Start Do Until loop   | None                       |
| ENDUNTIL | End Do Until loop     | Operator, value 1, value 2 |
| IF       | Start an IF statement | Operator, value 1, value 2 |
| ENDIF    | End IF statement      | None                       |
| FUNC     | Start function code   | Function name              |
| ENDFUNC  | End function code     | None                       |
| CALL     | Call function         | Function name              |
| PASSFAIL | Set pass or fail      | Pass or fail value         |
| ENDTEST  | End test              | None                       |

## GPIO

| Opcode  | Builder description            | Main arguments              |
|---------|--------------------------------|-----------------------------|
| SETPIN  | Set pin high or low            | Pin, level                  |
| RDPIN   | Read pin level                 | Pin, output register        |
| SETIN   | Set pin as input               | Pin                         |
| SETOUT  | Set pin as output              | Pin                         |
| MULTIO  | Set multiple pins as outputs   | Three pins                  |
| MULTII  | Set multiple pins as inputs    | Three pins                  |
| MULTIHI | Set multiple pins high         | Three pins                  |
| MULTILO | Set multiple pins low          | Three pins                  |
| ASAFE   | Set all pins to high impedance | None                        |
| ALLDIR  | Set all pin directions         | Side A and Side B bitfields |
| ALLEVEL | Set all pin levels             | Side A and Side B bitfields |
| PHIZ    | Set one pin to high impedance  | Pin                         |
| TOGPIN  | Toggle a pin                   | Pin, low-to-high flag       |

Safety: use ASAFE or appropriate input/high-impedance setup before changing connections. Do not command opposing outputs onto the same electrical node. Confirm the selected pin count because the on-screen package helper maps choices from that value.

## Timing and Memory

| Group  | Opcode                   | Purpose and arguments                                               |
|--------|--------------------------|---------------------------------------------------------------------|
| Timing | DELUS / DELMS            | Delay for microseconds or milliseconds.                             |
| Timing | CLKST / CLKSTP           | Start a clock on a pin at a selected kHz rate; stop the clock.      |
| Timing | BTNWT / BTNPR            | Wait for a button, or jump to a line when a button is pressed.      |
| Memory | DEFADDR                  | Define address pins using a base group plus low/high pin lists.     |
| Memory | SETADDR / GETADDR        | Set a 32-bit address or read an address into a register.            |
| Memory | REGLOD / REGINC / REGDEC | Load, increment, or decrement R0–R31.                               |
| Memory | ARRADD / ARRGET          | Write or read an array position.                                    |
| Memory | PIN2ARR                  | Build a pin array from lower and upper pin groups.                  |
| Memory | DEF8 / SET8 / GET8       | Define, drive, or read an indexed 8-bit group. Index range is 0–31. |

## Math

| Opcode    | Operation           | Arguments                                |
|-----------|---------------------|------------------------------------------|
| ADD       | Add                 | Left value, right value, output register |
| SUBTR     | Subtract            | Left value, right value, output register |
| MULT      | Multiply            | Left value, right value, output register |
| DIV       | Divide              | Left value, right value, output register |
| RNG       | Random number       | Minimum, maximum, output register        |
| AND       | Bitwise AND         | Left value, right value, output register |
| OR        | Bitwise OR          | Left value, right value, output register |
| XOR       | Bitwise XOR         | Left value, right value, output register |
| NOT       | Bitwise NOT         | Value, output register                   |
| SHL / SHR | Shift left or right | Value, shift amount, output register     |

## Add-on Boards

| Opcode       | Use                                         |
|--------------|---------------------------------------------|
| BRD_ON       | Turn on add-on-board power.                 |
| BRD_OFF      | Turn off add-on-board power.                |
| BRD_SEND_CMD | Send a command word plus two data values.   |
| BRD_GET_RSLT | Read result slot 0–63 into register R0–R31. |

The BRD\_SEND\_CMD helper explains the current identification commands: GET\_ID is command 1; name 2 and manufacturer 3 are string commands used by the board scanner. If Store result is on, the helper combines the command and chosen result slot. Data fields also accept #Rn or DIM values. Turn board power off when finished and confirm the board's electrical requirements before use.

## Display

| Opcode             | Use                                                                                |
|--------------------|------------------------------------------------------------------------------------|
| LEDON / LEDOFF     | Turn the indicator LED on or off.                                                  |
| BLINKON / BLINKOFF | Start blinking at a millisecond speed or stop blinking.                            |
| FILSCR             | Fill the screen with an RGB565 color.                                              |
| SHOWIMG            | Show a stored image by zero-based image index at X and Y.                          |
| SHOSTR             | Show String 0–7 at X and Y.                                                        |
| SEVSEG             | Draw a pseudo seven-segment number using packed position and size/color arguments. |
| TXTCOL             | Set foreground and background text colors.                                         |
| PROG               | Show a progress bar with percent, fill color, and background color.                |
| UDSCRN             | Refresh the display so queued drawing becomes visible.                             |

Current builder guidance for SHOWIMG: follow it with UDSCRN and DELMS 10. The same refresh pattern is useful after other screen drawing commands when the display must update immediately.

## 7 Strings and Display Messages

Each chip document contains eight text slots numbered 0 through 7. SHOSTR uses the slot number, so String 0 is addressed as 0, String 1 as 1, and so on. Unused strings may remain blank.

1. Enter the reusable text in a String field.
2. Insert SHOSTR and set Str Index, X, and Y.
3. Set colors first with TXTCOL if needed.

4. Add UDSCRN to refresh the screen.
5. Add DELMS if the message must remain visible before the next screen change.

```
SHOSTR 0x0 0x0 0x30
```

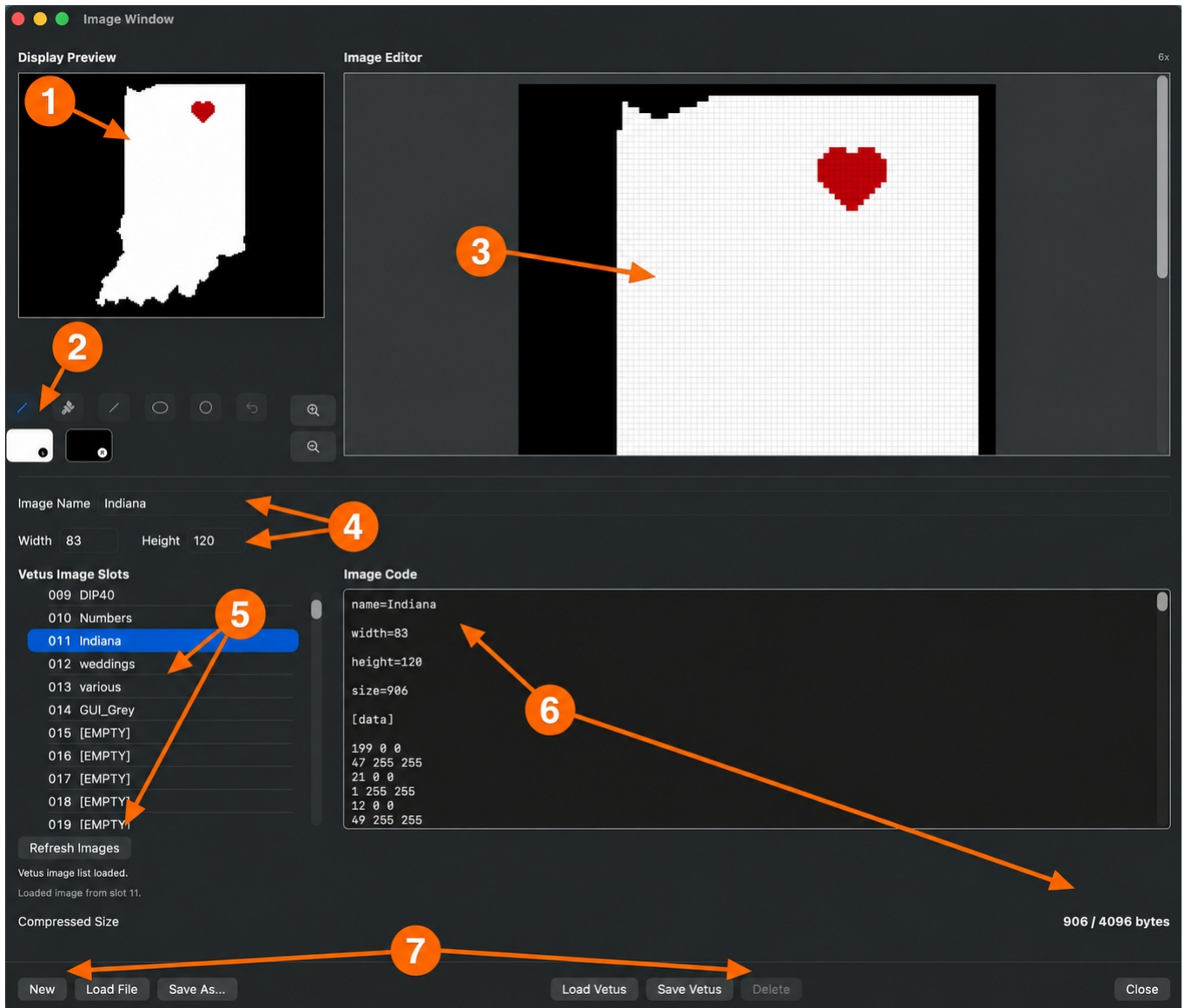
```
UDSCRN
```

```
DELMS 0xA
```

Formatting may include register substitutions, as shown in the supplied 4164 and 6502 files, for example row = %d #R0 and addr: %4X #R3. Keep the format marker and register reference in the same string. Because firmware performs the final formatting, verify the result on Vetus when using widths or hexadecimal output.

## 8 Image Manager

Open Tools then Show Image Window, or press Command Shift I. Images can be made and edited locally, saved as text files, and loaded from or saved to Vetus image slots.



| No. | Area            | Use                                                     |
|-----|-----------------|---------------------------------------------------------|
| 1   | Display Preview | Shows the image at display scale on a black background. |

| No. | Area                   | Use                                                                                               |
|-----|------------------------|---------------------------------------------------------------------------------------------------|
| 2   | Drawing controls       | Pencil, paint bucket, line, ellipse, circle, undo, left-click color, right-click color, and zoom. |
| 3   | Image Editor           | Pixel grid. Drag or click with the selected tool; use left or right mouse color.                  |
| 4   | Image information      | Image name, width, and height.                                                                    |
| 5   | Vetus Image Slots      | Select a slot and refresh the stored-image list.                                                  |
| 6   | Image Code and size    | Shows generated text and compressed usage. Maximum compressed size is 4096 bytes.                 |
| 7   | File and Vetus actions | New, Load File, Save As, Load Vetus, Save Vetus, Delete, and Close.                               |

## Create or edit an image

1. Click New. Enter a name of 31 UTF-8 bytes or fewer and choose a size from 1 by 1 up to 160 by 128.
2. Choose Pencil, Paint Bucket, Line, Ellipse, or Circle.
3. Choose separate left-click and right-click colors. VetusGUI converts drawing colors to RGB565.
4. Draw in the grid. Use Undo Last Edit to step back; zoom ranges from 2× to 20×.
5. Watch Compressed Size. Save Vetus is disabled above 4096 bytes. Large areas of one color compress better than noisy pixel patterns.
6. Click Save As to keep an editable text copy on the Mac.

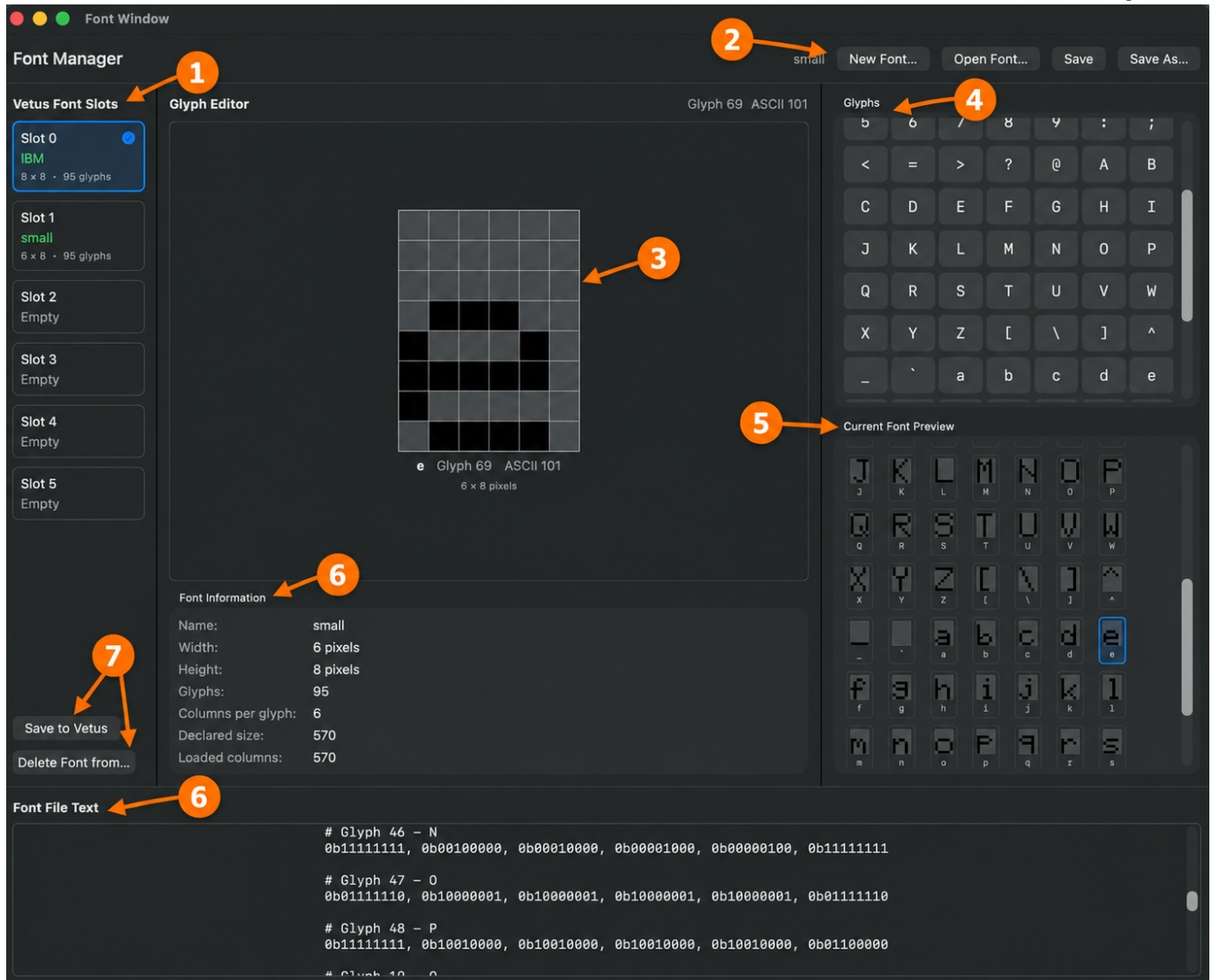
## Move images to and from Vetus

1. Connect Vetus and click Refresh Images.
2. Select a numbered slot.
3. Click Load Vetus to copy an occupied slot into the editor, or Save Vetus to write the current image into the selected slot.
4. Use Delete only when you intend to empty the selected slot. Current behavior protects slots 1 through 14 from deletion; Delete is enabled for occupied slot 15 or higher.
5. Use the image's zero-based index with SHOWIMG in chip code.

Saving to a slot may ask before replacing existing content. Deleting a slot does not renumber other images.

## 9 Font Manager

Open Tools then Show Font Window, or press Command Shift F. The manager creates bitmap fonts, edits individual glyph pixels, saves font text files, and transfers fonts to one of six Vetus font slots.



| No. | Area                      | Use                                                                             |
|-----|---------------------------|---------------------------------------------------------------------------------|
| 1   | Vetus Font Slots          | Select slot 0 through 5. Occupied slots show name, dimensions, and glyph count. |
| 2   | File controls             | New Font, Open Font, Save, and Save As.                                         |
| 3   | Glyph Editor              | Click cells to turn pixels on or off for the selected glyph.                    |
| 4   | Glyphs                    | Choose the character to edit. Glyph 0 corresponds to ASCII 32.                  |
| 5   | Current Font Preview      | Review the whole font and jump to a glyph.                                      |
| 6   | Information and file text | Check dimensions/counts and view the generated editable text.                   |
| 7   | Device controls           | Save the current font to the selected slot or delete that slot.                 |

## Create and edit a font

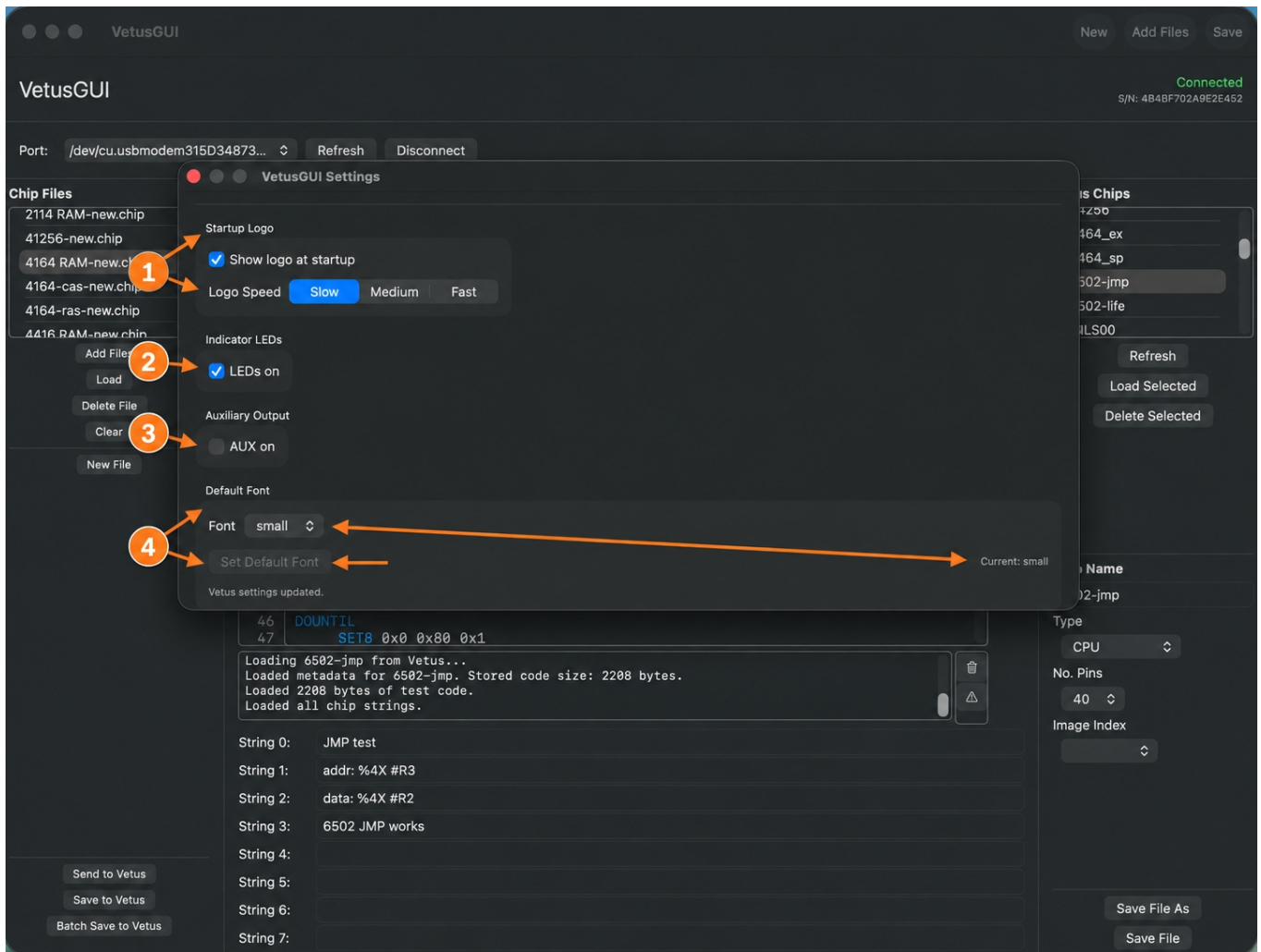
1. Click New Font. Enter a name, width from 1 to 32 pixels, height from 1 to 8 pixels, and 1 to 224 glyphs.
2. Choose a glyph from the right. Standard 95-glyph fonts cover ASCII 32 through 126.
3. Click cells in the center grid to toggle pixels. The preview and Font File Text update immediately.
4. Click Save. A new font uses Save As the first time; later Save writes to the same file. Font files are UTF-8 plain text.

## Load and store fonts

1. Open Font loads a Vetus font text file and validates its name, width, height, size, data values, column height, and complete glyph width.
2. Connect Vetus and select one of slots 0 through 5.
3. Click Save to Vetus. Fonts taller than 8 pixels are rejected.
4. Click Delete Font from Slot only when you want to empty the selected device slot.
5. Choose the device's normal font in Settings after it has been saved to Vetus.

## 10 Settings

Open the VetusGUI Settings window while Vetus is connected. The app requests the current settings and font list when the window appears.



| No. | Setting          | What it changes                                                                             |
|-----|------------------|---------------------------------------------------------------------------------------------|
| 1   | Startup Logo     | Show or hide the startup logo. When enabled, choose Slow, Medium, or Fast animation speed.  |
| 2   | Indicator LEDs   | Turns the Vetus indicator LEDs on or off.                                                   |
| 3   | Auxiliary Output | Turns the AUX output on or off.                                                             |
| 4   | Default Font     | Selects a non-empty Vetus font slot as the default. The current font is shown at the right. |

Controls are disabled while disconnected or while a settings update is in progress. Set Default Font is also disabled when the selected font is already current. Status and error text appears at the bottom of the Default Font section.

## 11 File Formats and Limits

| Item           | Current limit or behavior                                                                              |
|----------------|--------------------------------------------------------------------------------------------------------|
| Chip file      | UTF-8 text in .chip format. One chip per file, with metadata, code, and strings 0–7.                   |
| Chip types     | Unknown, DRAM, SRAM, CPU, 7400 Series, I/O, Other. Unknown cannot be transferred as a finished chip.   |
| Chip pins      | Even choices 2 through 40.                                                                             |
| Package images | DIP6, DIP14, DIP16, DIP20W, DIP20, DIP24, DIP28, DIP32, DIP40.                                         |
| Registers      | R0 through R31; use #Rn when an argument should read a register value.                                 |
| Strings        | Eight per chip, indexes 0 through 7.                                                                   |
| Images         | 1–160 pixels wide, 1–128 high; name at most 31 UTF-8 bytes; compressed device data at most 4096 bytes. |
| Image files    | UTF-8 text with name, width, height, size, [data], and run-length RGB565 byte triples.                 |
| Fonts          | Six device slots. New fonts: width 1–32, height 1–8, glyph count 1–224.                                |
| Font files     | UTF-8 text with name, width, height, size, [data], then binary column values.                          |
| Address groups | Address 1–8, 9–16, 17–24, or 25–32; indexed 8-bit groups use indexes 0–31.                             |
| Add-on results | Result slot 0–63 into output register 0–31.                                                            |

Forward compatibility: the chip-file reader ignores unknown metadata keys, but it rejects missing required sections and invalid names, pins, types, or image values.

## 12 Troubleshooting and Safe Practices

| Problem                                 | What to check                                                                                         |
|-----------------------------------------|-------------------------------------------------------------------------------------------------------|
| No port or cannot connect               | Use a data USB cable, power Vetus, click Refresh, choose the proper port, and try Connect again.      |
| Device buttons are gray                 | Connect first. Some actions also require a selected occupied slot, a loaded document, or valid data.  |
| Send or Save stops                      | Read the terminal. Supply chip name/type/pins, then correct compiler errors.                          |
| Stored chip list looks old              | Click Refresh under Vetus Chips.                                                                      |
| Image will not save to Vetus            | Check connection, name, dimensions, selected slot, and compressed size at or below 4096 bytes.        |
| Font will not upload                    | Select a slot, load a font, and keep height at 8 pixels or less.                                      |
| Default font list is empty              | Save at least one font to Vetus, then reopen or refresh Settings.                                     |
| Loaded old code changes condition names | VetusGUI normalizes compatible condition spellings. Re-run the error check and save a new local copy. |
| Unexpected screen output                | Check string/image index, X/Y values, colors, UDSCRN, and a short DELMS after SHOWIMG.                |
| Transfer interrupted                    | Reconnect, refresh the relevant list, confirm what reached Vetus, then retry the single item.         |

## Electrical and storage safety

- Remove power before inserting or removing a chip or add-on board unless the hardware procedure explicitly says otherwise.
- Match the chip orientation, package, pin count, and voltage requirements.
- Begin uncertain GPIO sequences in a safe input or high-impedance state.
- Treat Delete Selected, Delete All Chips, Delete Image, and Delete Font as destructive device actions. Save local copies first.
- Watch the terminal until each transfer finishes. Do not unplug Vetus during a write.

## Appendix Quick Reference

| Goal                      | Shortest path                                                                      |
|---------------------------|------------------------------------------------------------------------------------|
| Start a chip              | New → name/type/pins/image → code/strings → Save File As                           |
| Open local test           | Add Files → select file → Load                                                     |
| Run current test          | Connect → Send to Vetus                                                            |
| Store current test        | Connect → Send to Vetus → Save to Vetus                                            |
| Copy many local tests     | Add Files → Batch Save to Vetus                                                    |
| Edit a stored test        | Refresh Vetus Chips → select → Load Selected → edit → save locally and/or to Vetus |
| Check code                | Warning triangle beside terminal                                                   |
| Insert an opcode          | Choose category icon → action → arguments → Insert                                 |
| Manage images             | Tools → Show Image Window                                                          |
| Manage fonts              | Tools → Show Font Window                                                           |
| Change device preferences | VetusGUI Settings                                                                  |
| Back up work              | Save .chip, image .txt, and font .txt files on the Mac                             |