

## Examples: Version 10.0

This is a collection of diagrams the author has had occasion to produce using m4 circuit macros and others, and dpic or gpic. In some cases there are other or better m4 or pic constructs for producing the same drawings, but names of the actual source-files are shown for reference. Some of the later examples test the boundaries of what can be done when employing a “little language” like pic. Most of the examples can be processed using either dpic -p, dpic -g, or, with exceptions, gpic -t, but the possibility of other postprocessing has meant that sometimes the source is slightly more complicated than it would be if only one workflow had been assumed. The most simplicity and elegance is achieved by sticking to one pic interpreter and one postprocessor.

This document duplicates a few diagrams from the manual Circuit\_macros.pdf. There are also a few files in the examples directory that are not included in this document. To process *file.m4*, for example, type “make *file.pdf*”.

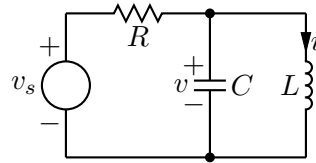


Figure 1: The quick-start example from the manual [quick.m4].

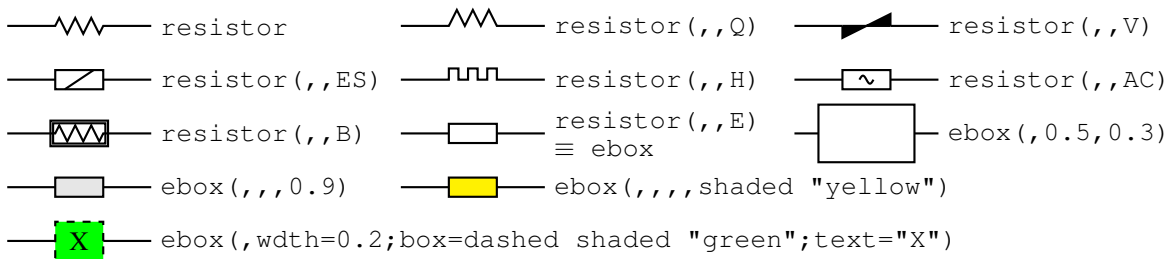


Figure 2: Resistors, showing some variations and the ebox [Resistors.m4].

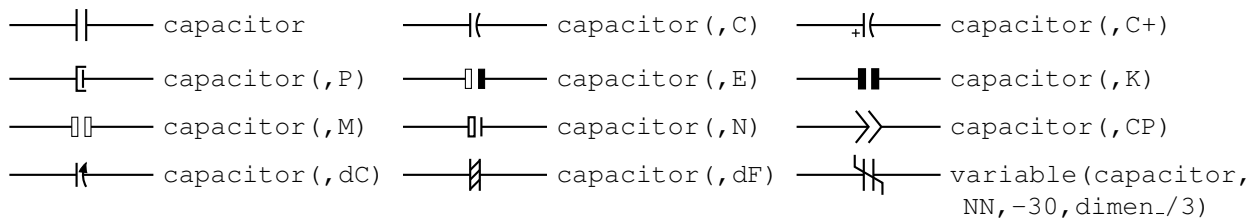


Figure 3: Capacitors [Capacitors.m4].

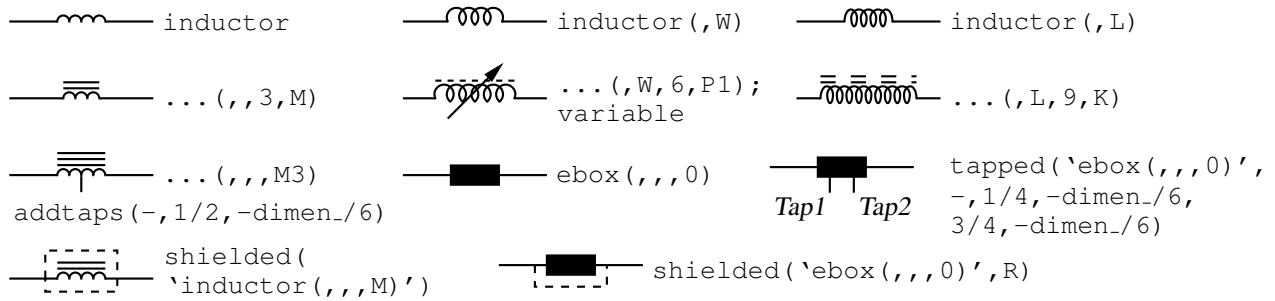


Figure 4: Inductors [Inductors.m4].

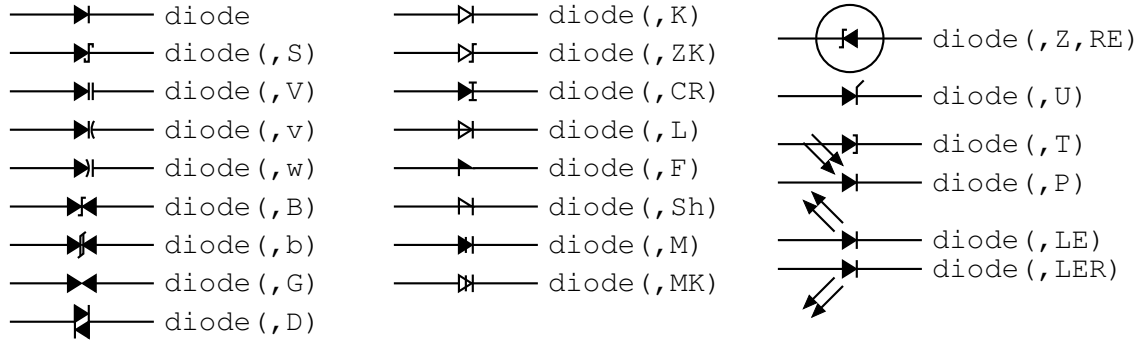


Figure 5: Diodes: appending a K to the second argument draws an open arrowhead [Diodes.m4].

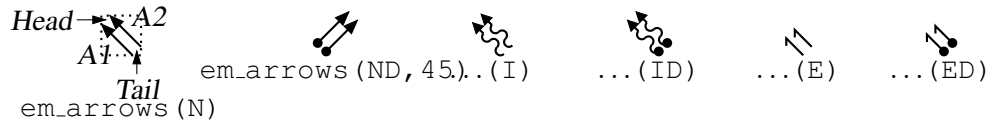


Figure 6: Radiation arrows [Emarrows.m4].

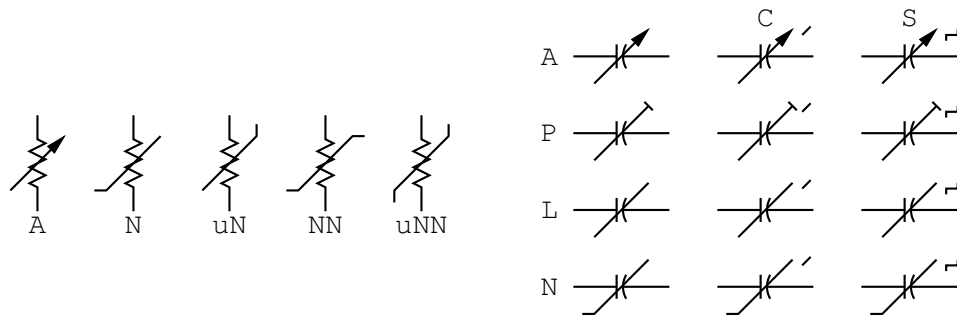


Figure 7: Arrows and marks for showing variability [Variable.m4].

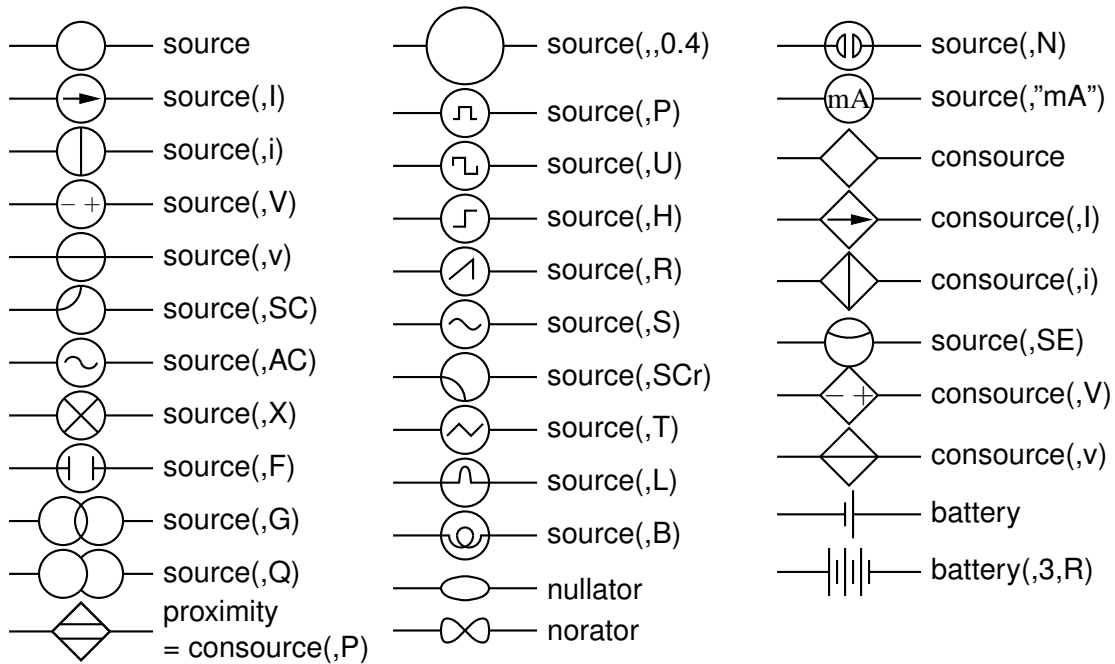


Figure 8: Sources and source-like elements [Sources.m4].

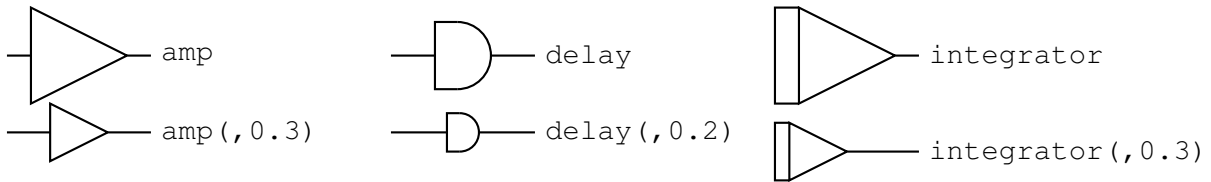


Figure 9: Macros amp, delay, and integrator [AmpTable.m4].

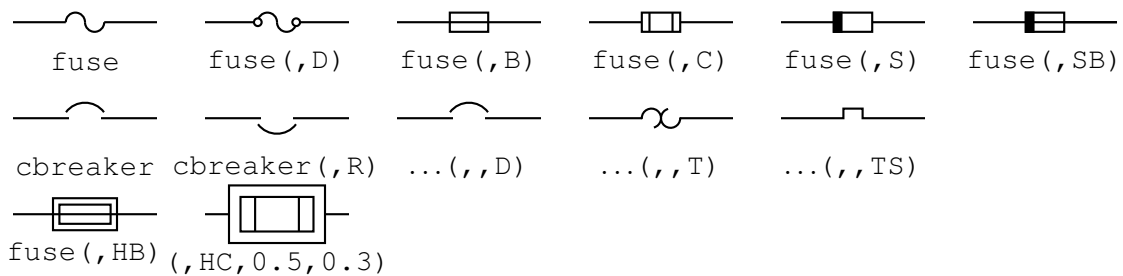


Figure 10: Macros fuse and cbreaker [Fuses.m4].





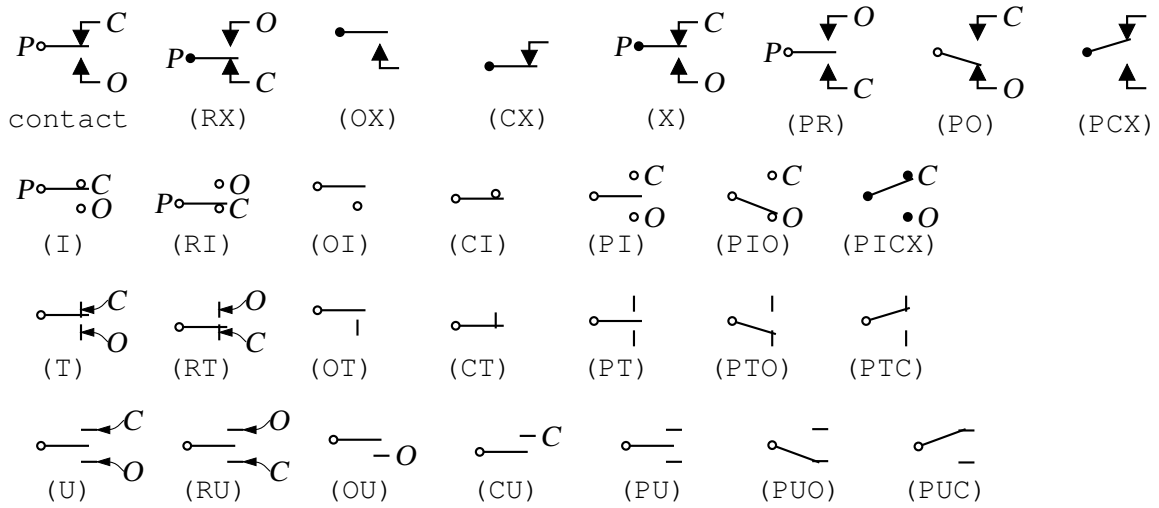


Figure 19: A non-exhaustive sampling of contact macro variations [Contact.m4].

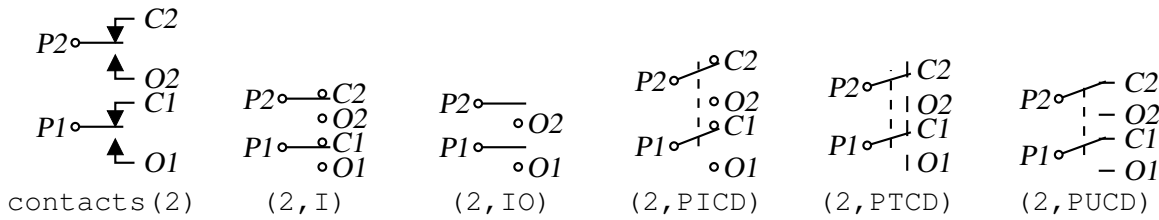


Figure 20: The contacts macro [Contacts.m4].

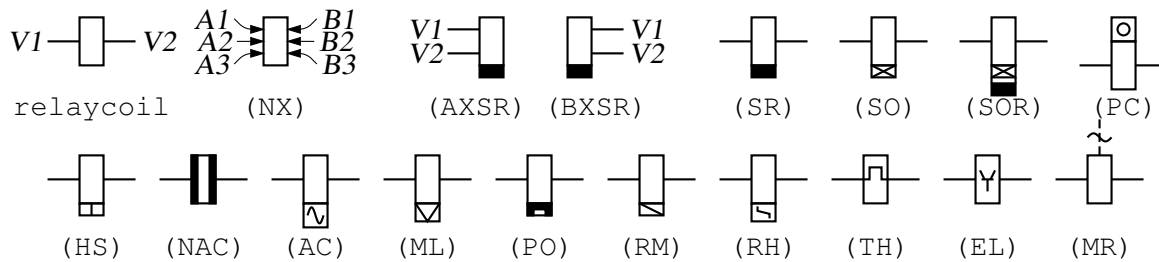


Figure 21: The relaycoil macro [relaycoil.m4].

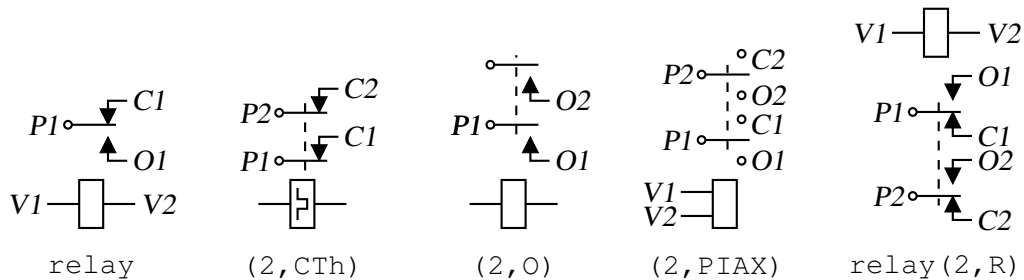


Figure 22: Some variants of relay [Relay.m4].

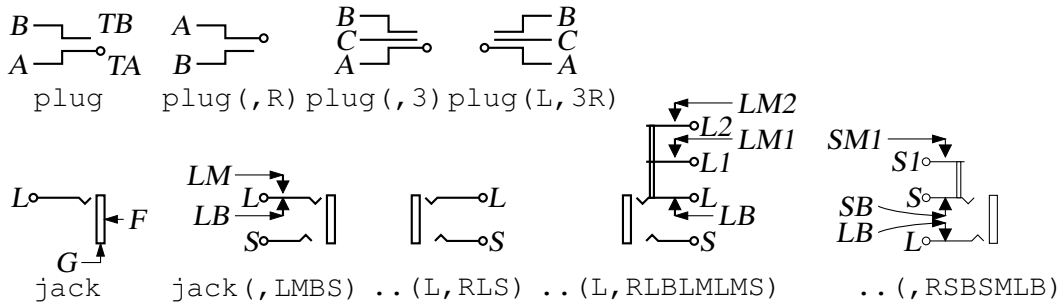


Figure 23: The jack and plug macros [Jack.m4].

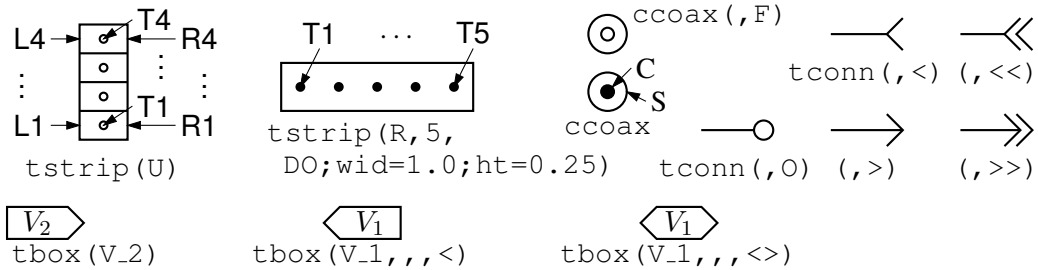


Figure 24: The tstrip, ccoax, tconn, and tbox macros [Conn.m4].

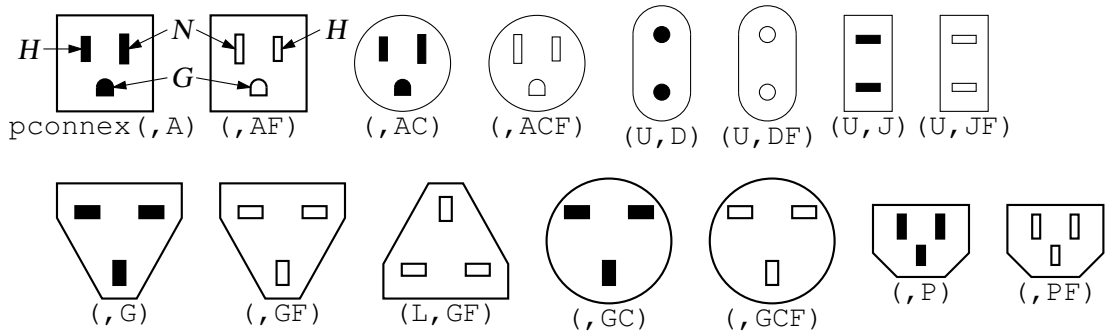


Figure 25: The pconnex macro [Pconn.m4].

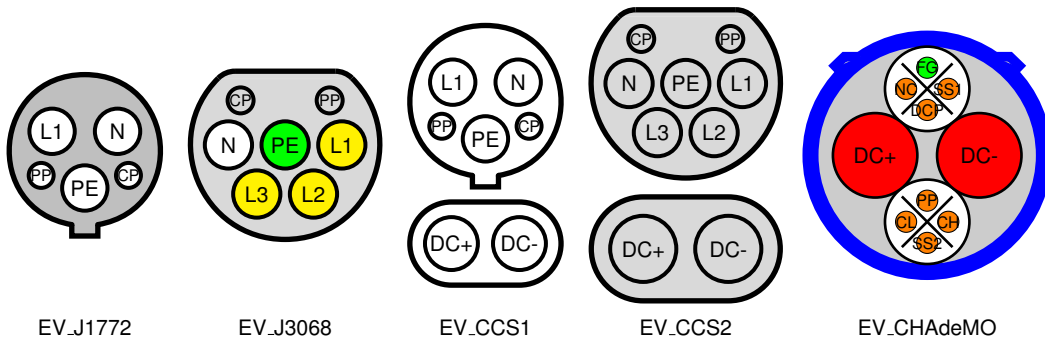


Figure 26: Electric vehicle charging plug patterns make extensive use of key=value pairs to set options [EVplugs.m4].

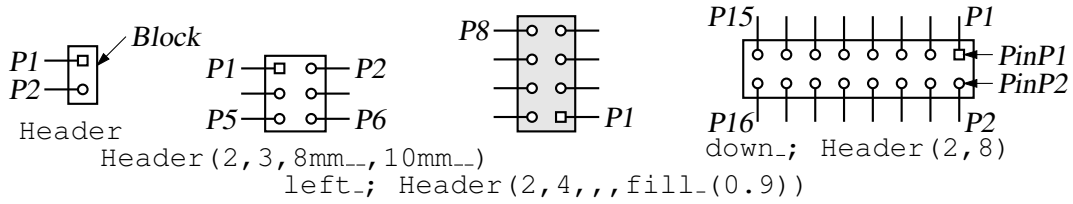


Figure 27: The Header macro [Headers.m4].

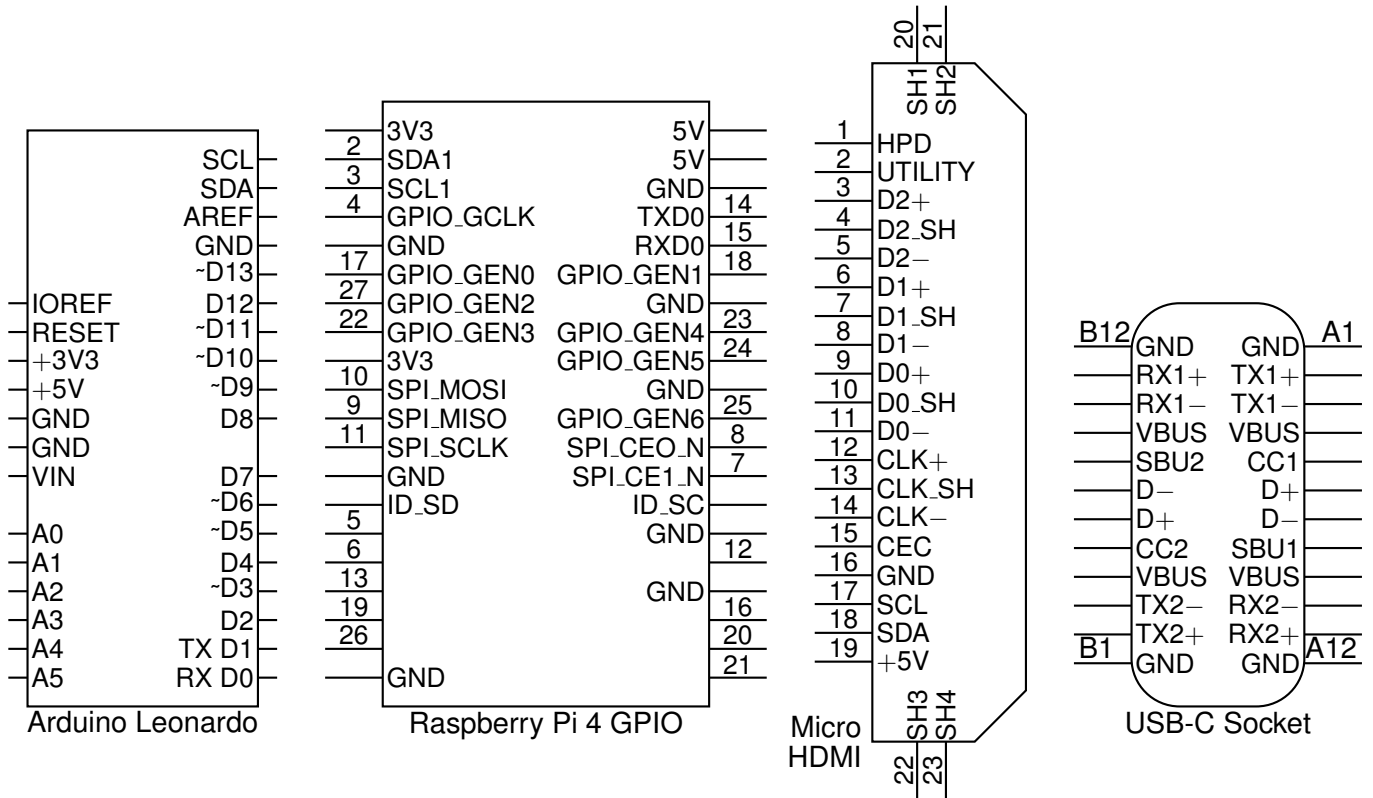


Figure 28: Some connectors with simple geometry and lists of labels [Connectors.m4].



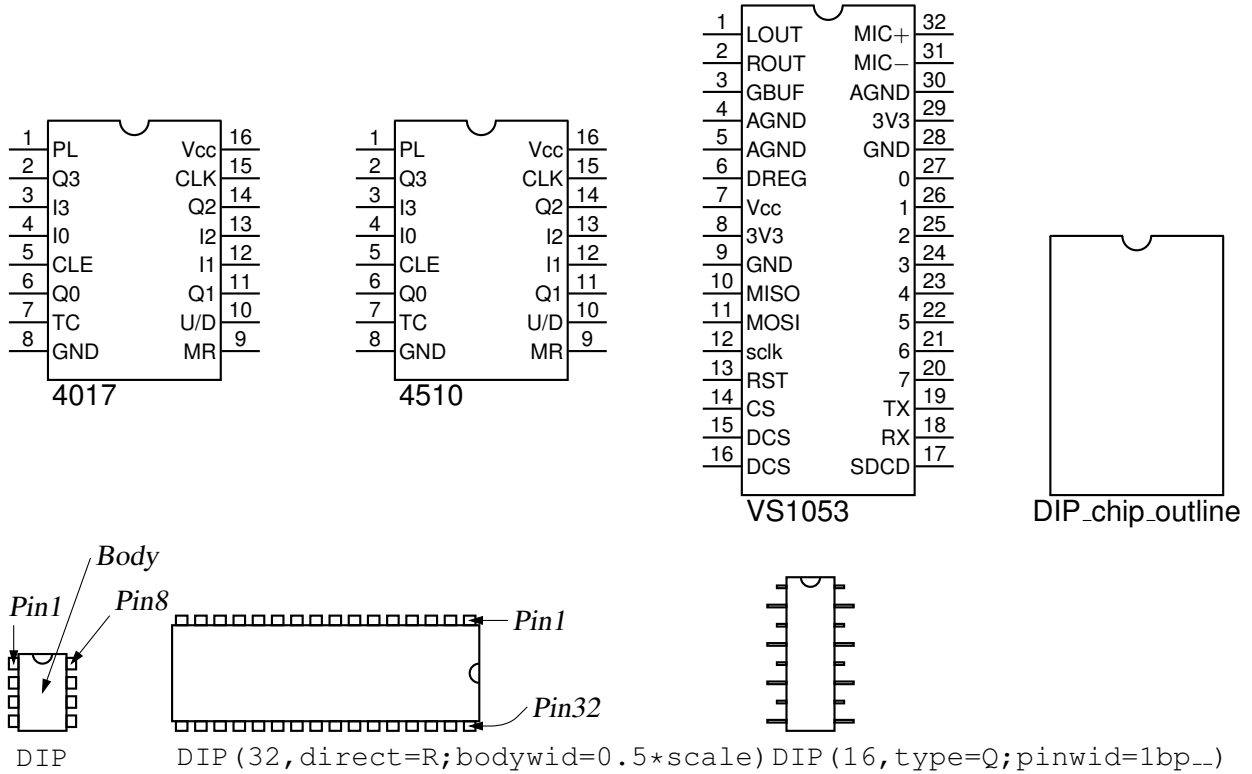


Figure 29: IC package outlines [Chips.m4].

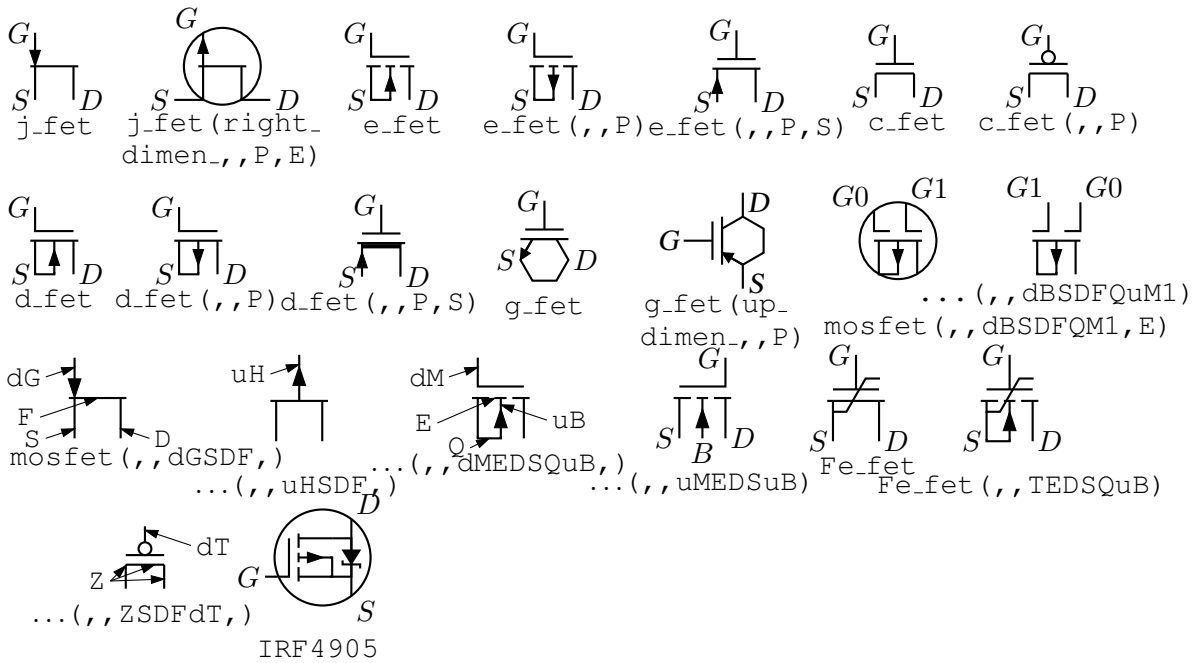


Figure 30: FETs, showing programmable components and example customizations [fet.m4].

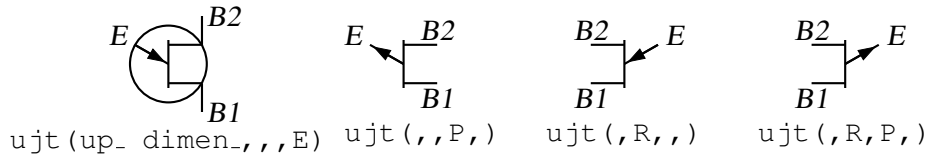


Figure 31: UJT examples [ujt.m4].

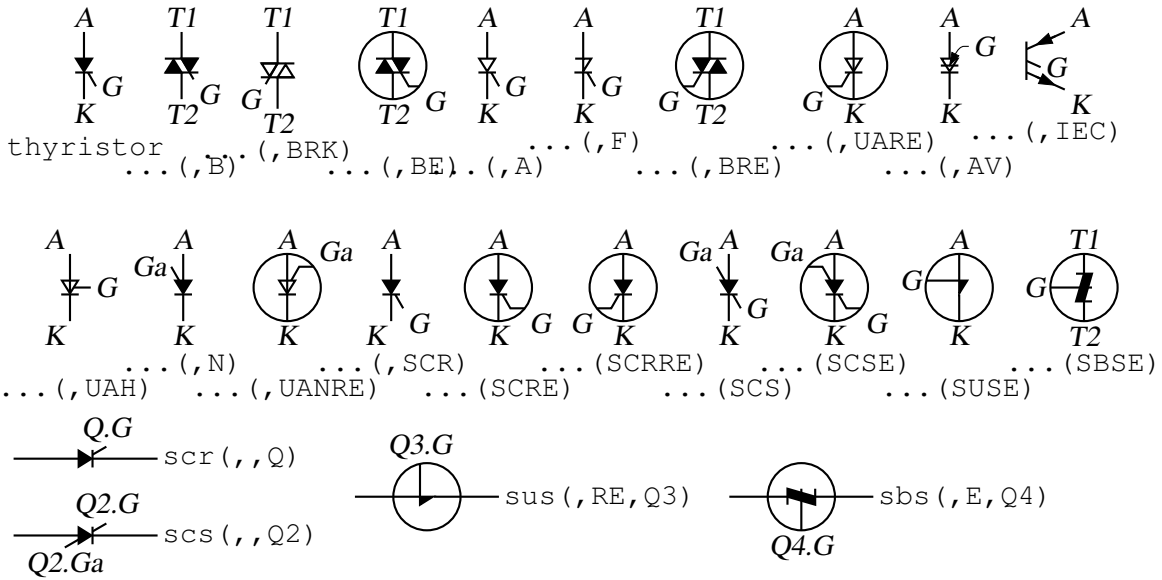


Figure 32: Thyristor examples. The thyristor is a 3- or 4-terminal composite element [thyristor.m4].

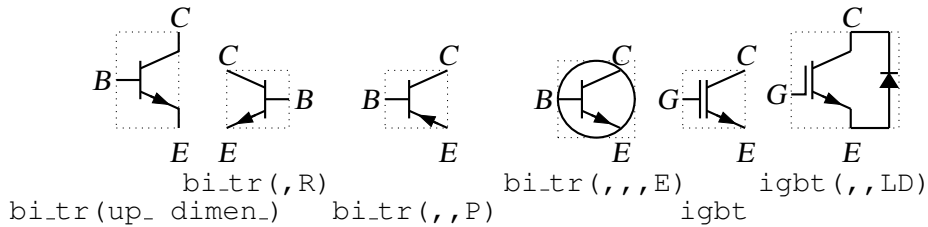


Figure 33: Bipolar transistors (drawing direction: up) [Bip.m4].

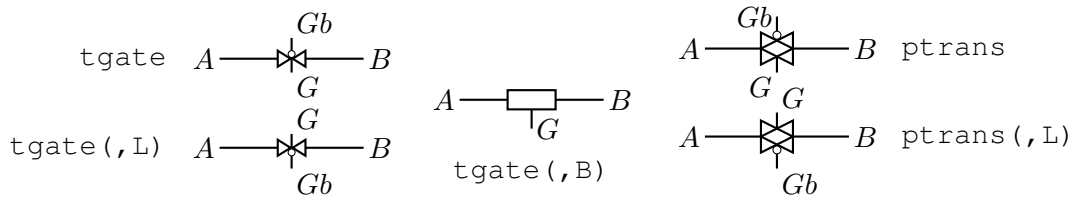


Figure 34: The tgate and ptrans elements [Tgate.m4].

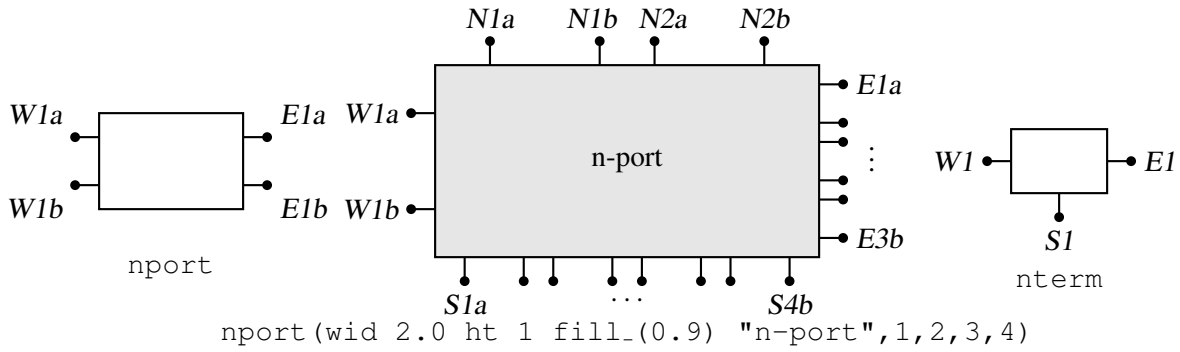


Figure 35: The nport and nterm macros [Nport.m4].

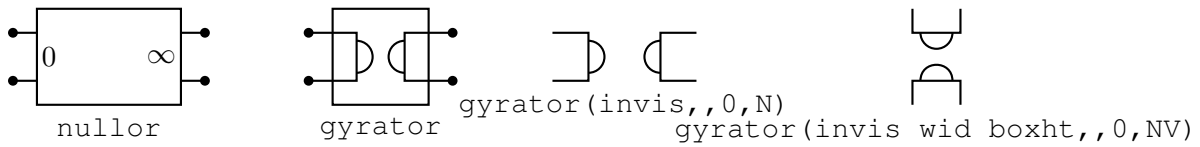


Figure 36: Some customizations of nport [NLG.m4].

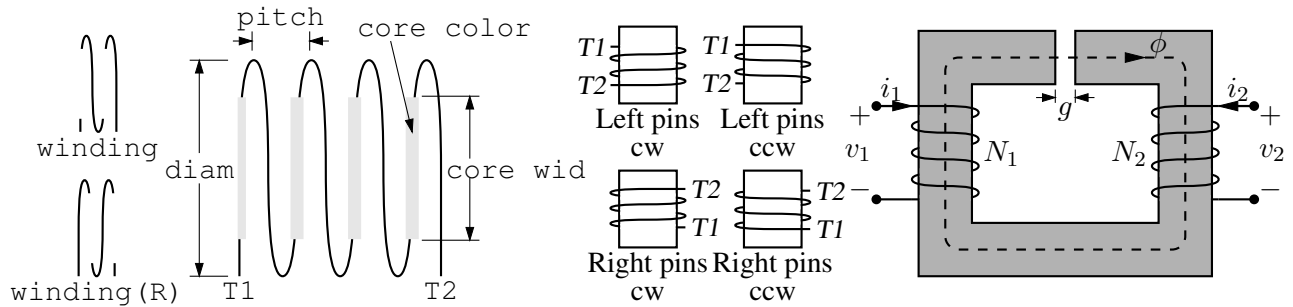


Figure 37: The macro winding(L|R,diam,pitch,turns,core wid,core color) [Windings.m4].

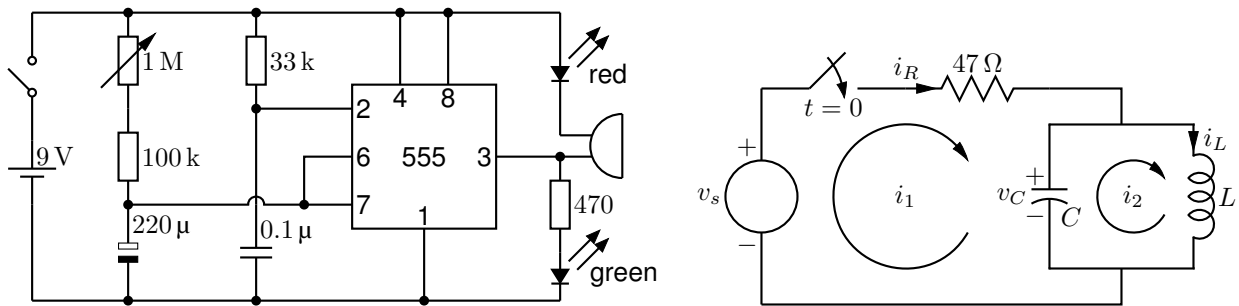


Figure 38: Two simple labeled circuits [ex01.m4].

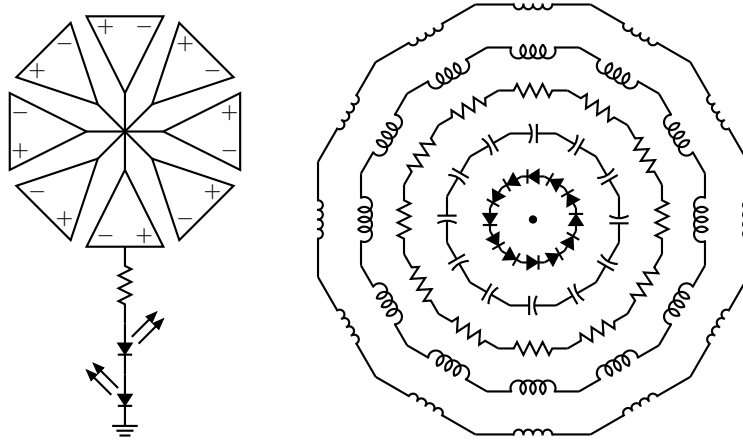


Figure 39: Elements at obtuse angles [ex02.m4].

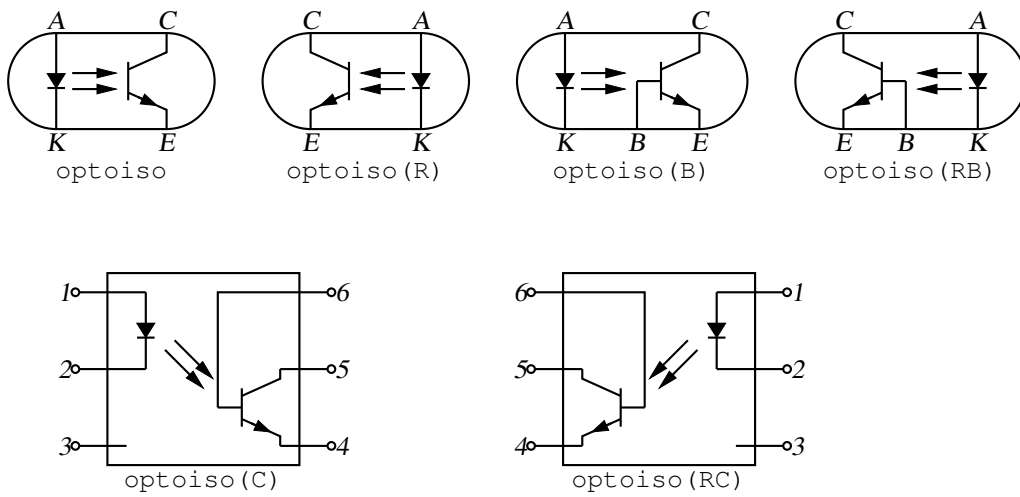


Figure 40: Optical isolator: a circuit with right or left orientation [Optoiso.m4].

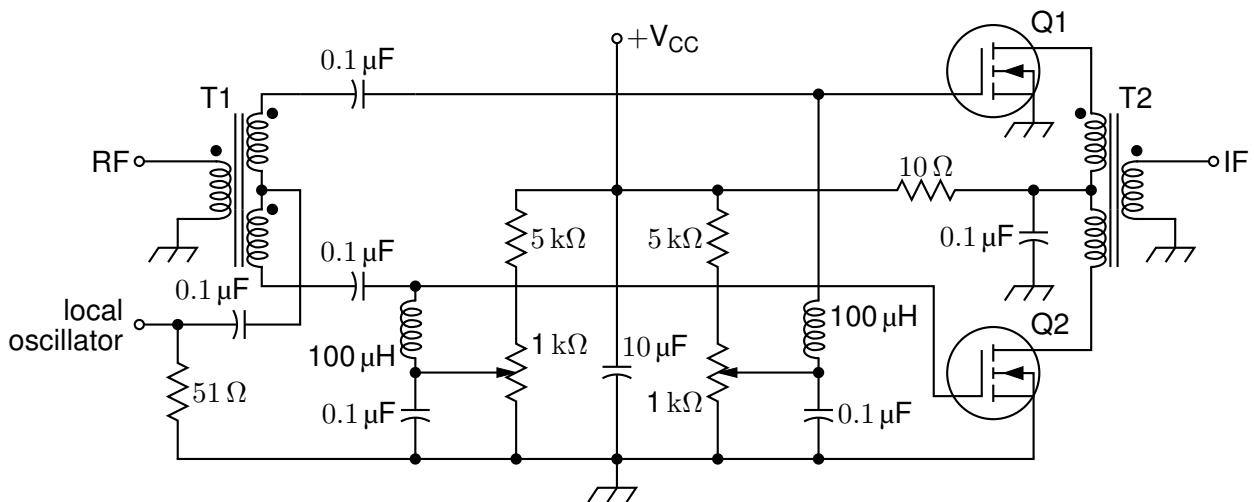


Figure 41: A balanced mixer, using mosfet and a custom transformer [Mixer.m4].

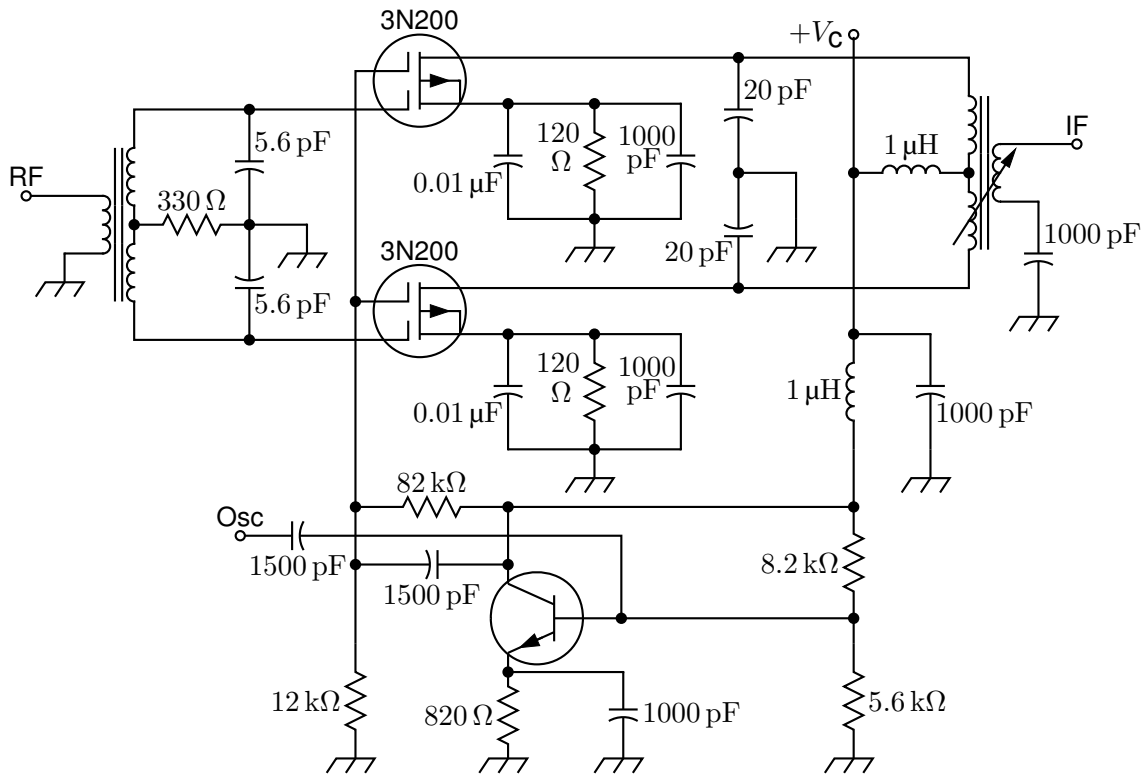


Figure 42: A push-pull mixer, showing FETs with multiple gates [PushPull.m4].

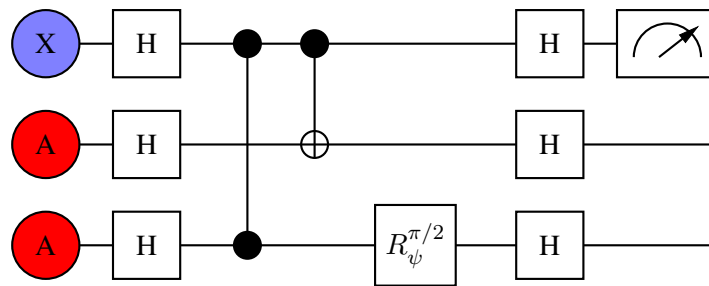


Figure 43: A quantum circuit [Quantum.m4].

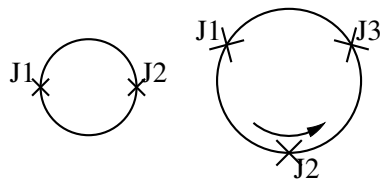


Figure 44: Superconducting quantum interface device (drawing direction down) [SQUID.m4].

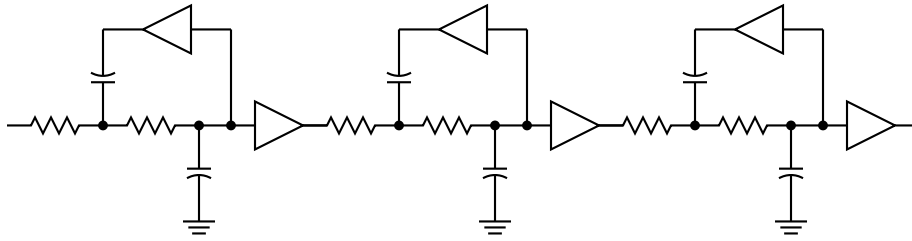


Figure 45: A six-pole filter [Sixpole.m4].

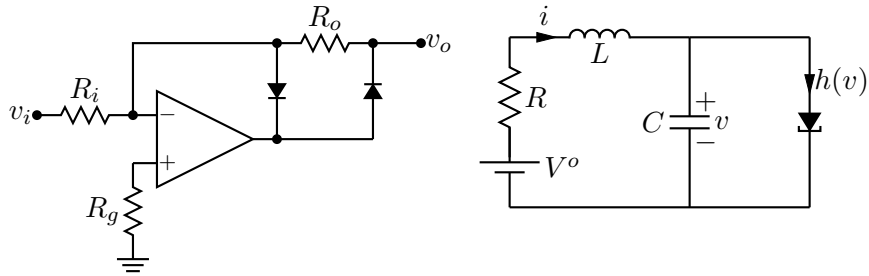


Figure 46: Precision half-wave rectifier and a tunnel diode circuit (illustrating opamp, diode, resistor, ground, and labels) [ex18.m4].

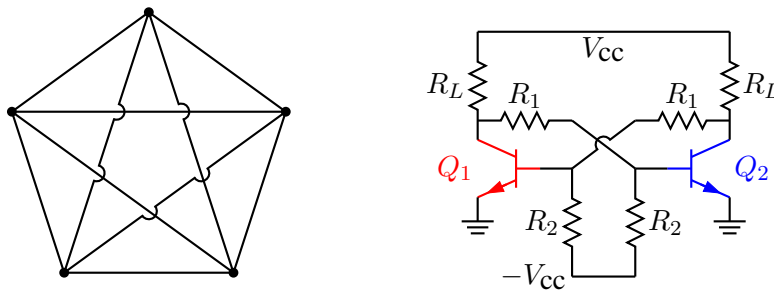


Figure 47: Non-planar graph and bistable circuit (illustrating the crossover macro and colored elements) [ex10.m4].

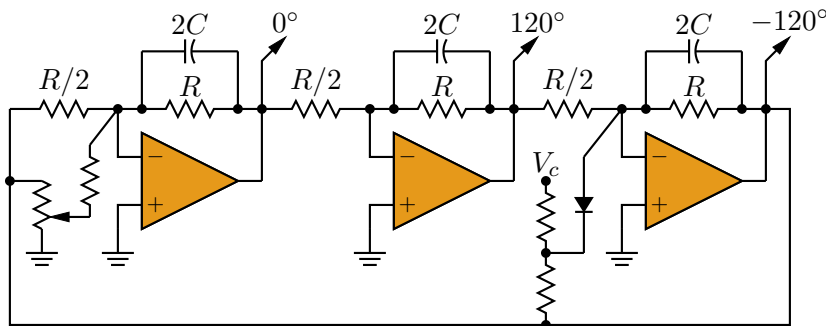


Figure 48: Three-phase oscillator [Three.m4].

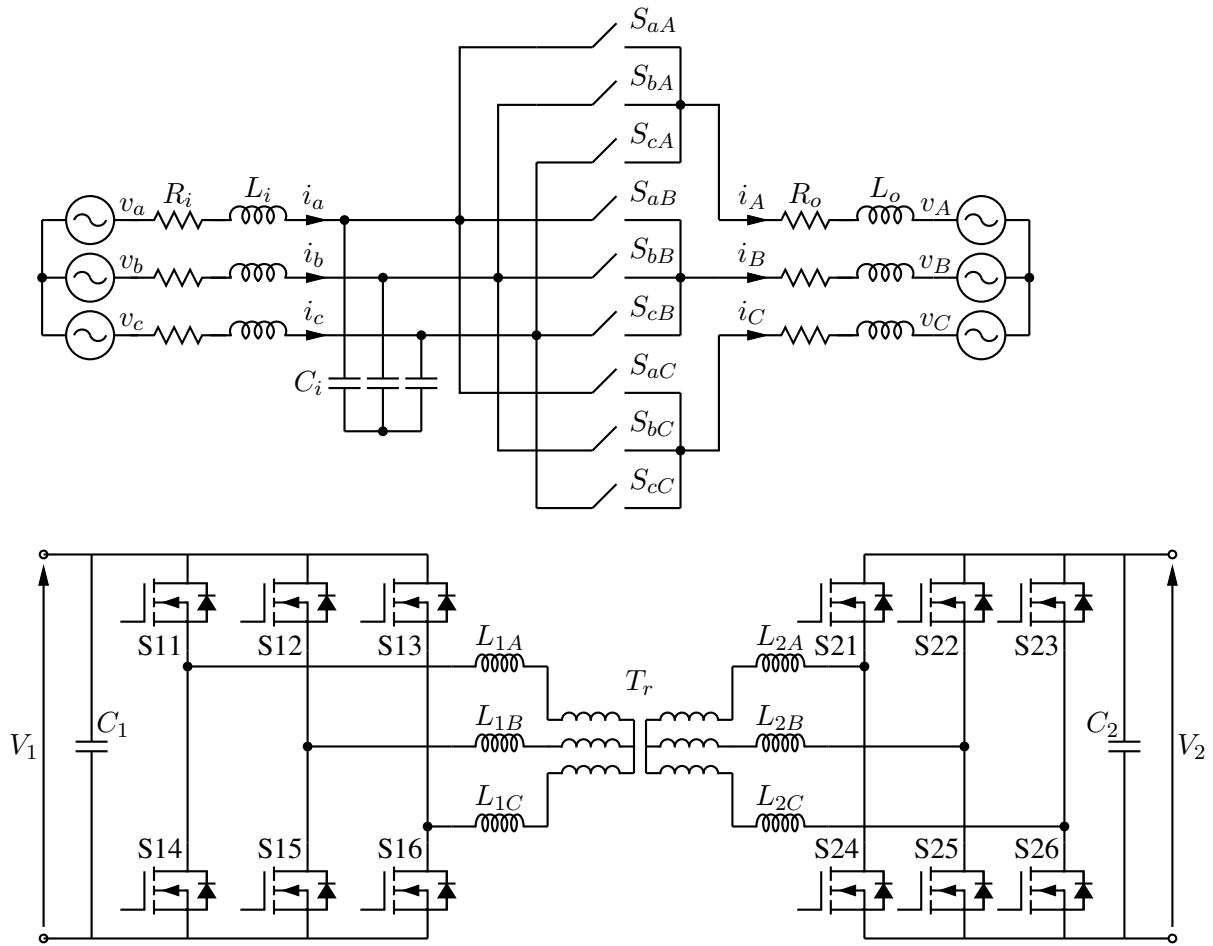


Figure 49: A three-phase switched AC-AC converter and a DC-DC converter [MC.m4].

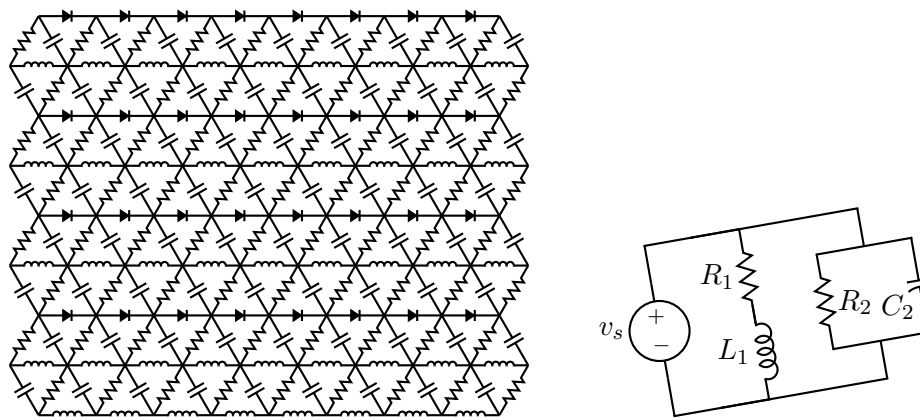


Figure 50: A repetitive network created by Pic looping and a skewed circuit used to test the macro parallel. [ex17.m4].

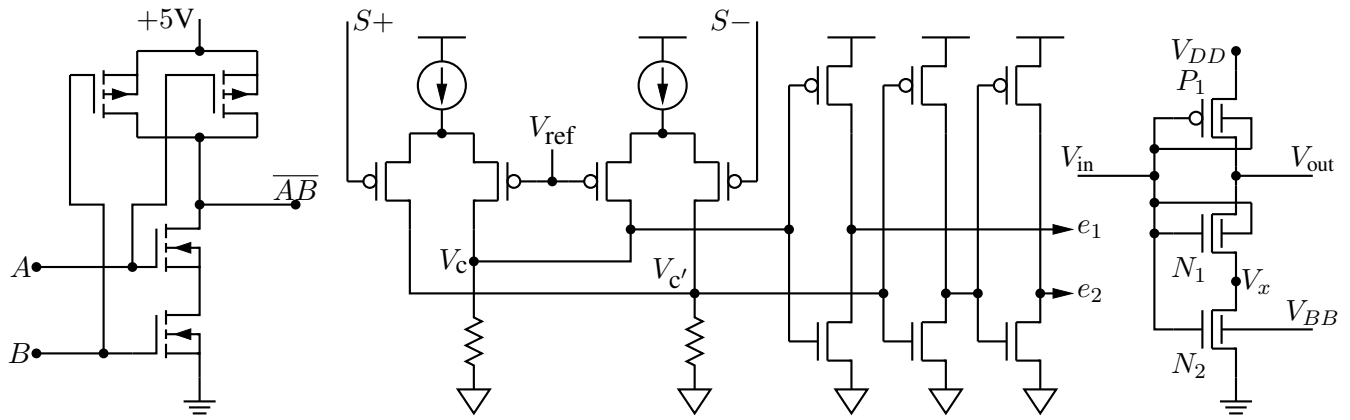


Figure 51: A CMOS NAND gate, a test circuit, and an X MOSFET example [ex12.m4].

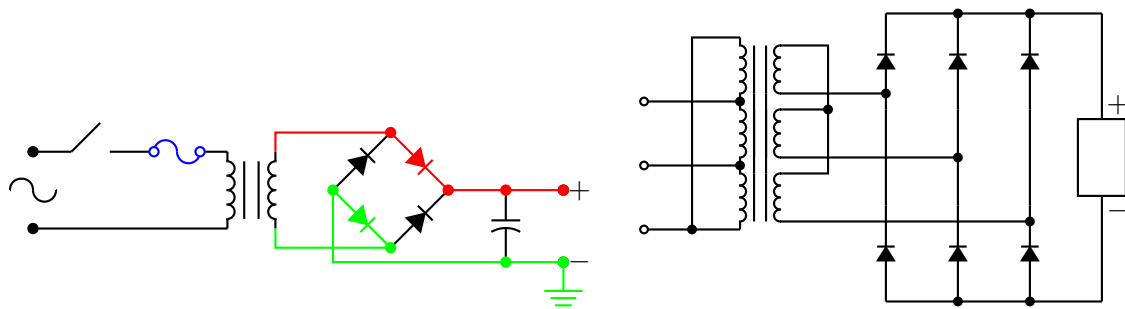


Figure 52: An elementary power supply circuit with colored elements, and a multiple-winding transformer with 3-phase rectifier [pwrsupply.m4].

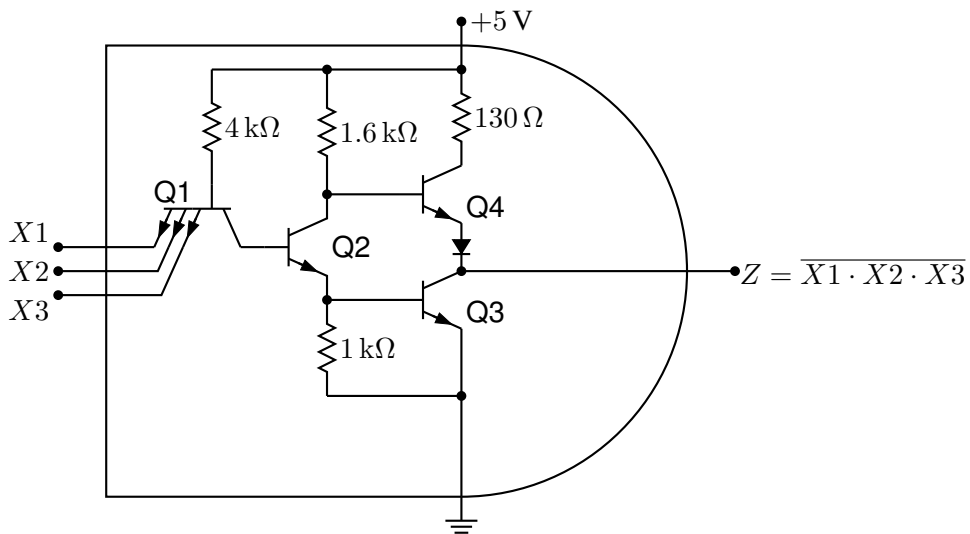


Figure 53: TTL NAND gate illustrating a transistor with multiple emitters [TTLnand.m4].



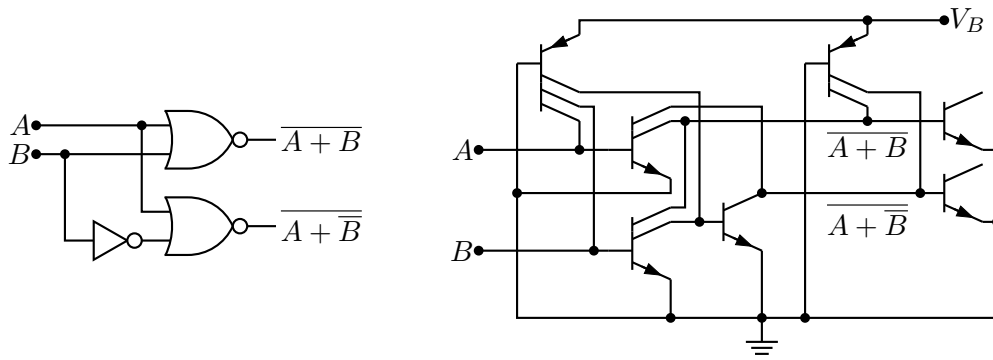


Figure 54: Gate circuit and equivalent embedded  $I^2L$  components illustrating multiple collectors [I2L.m4].

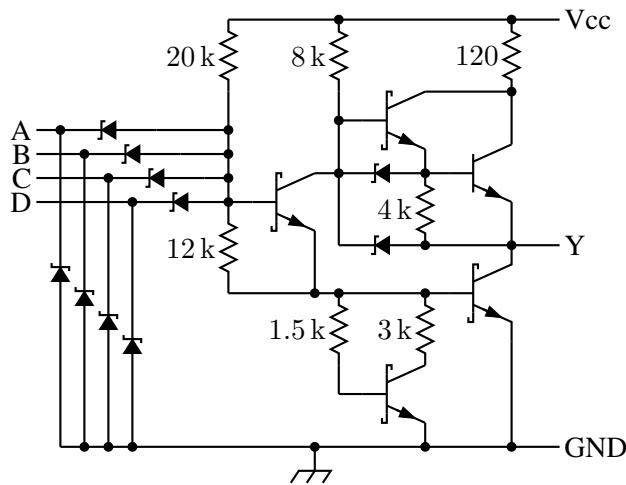


Figure 55: A 4-input NAND circuit illustrating the S (Schottky) option of bi.trans [Schottky.m4].

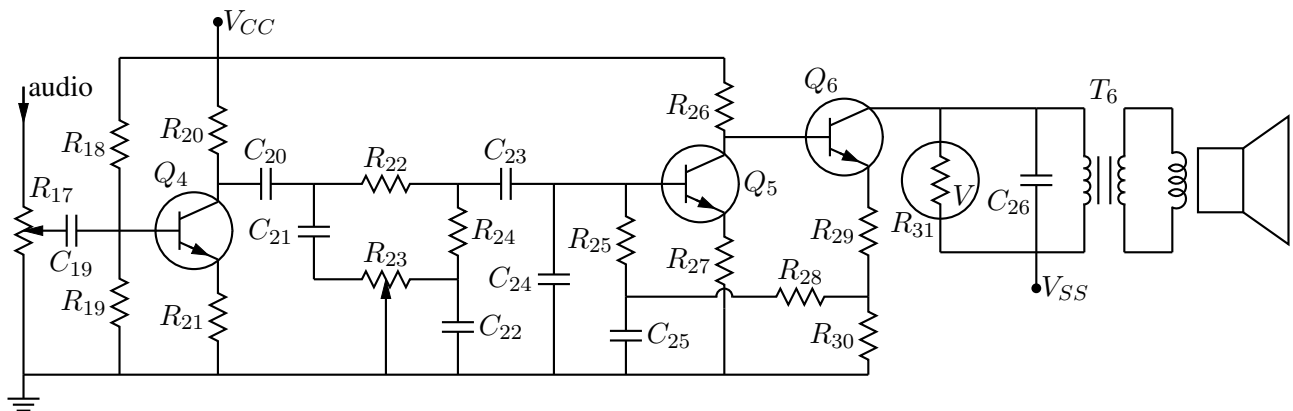


Figure 56: Transistor radio audio chain [ex11.m4].

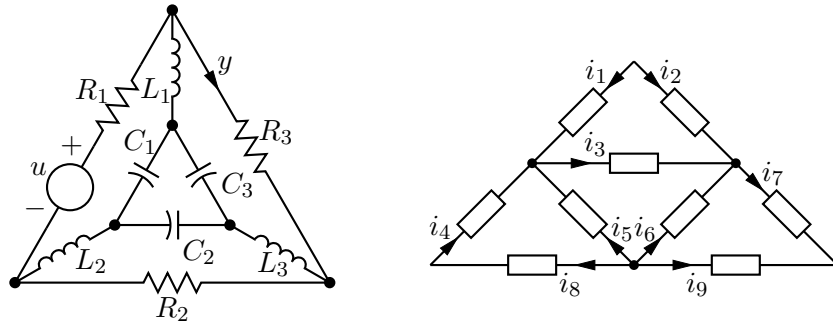


Figure 57: Labels on non-manhattan elements [ex04.m4].

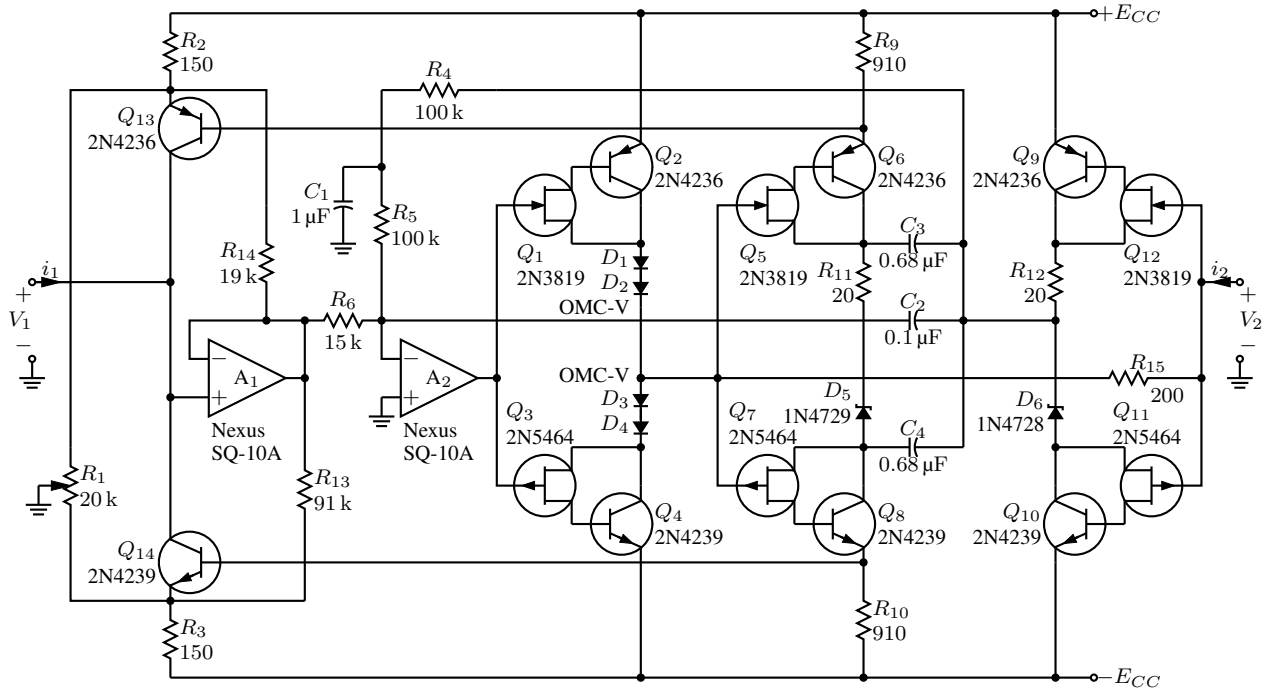


Figure 58: Realization of a controlled source (illustrating stacked element labels) [Csource.m4].

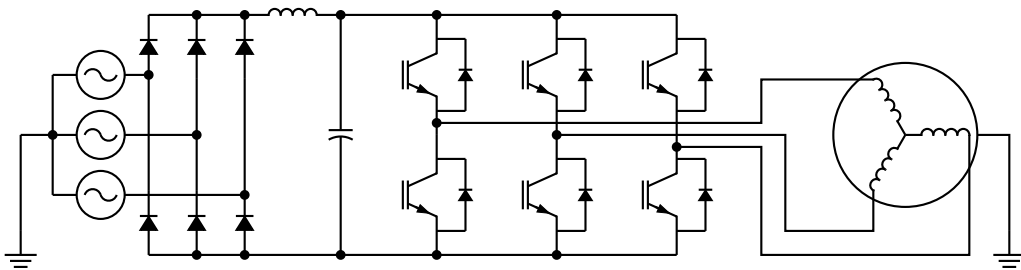


Figure 59: Synchronous machine driven by variable-speed drive and rectifier [Drive.m4].

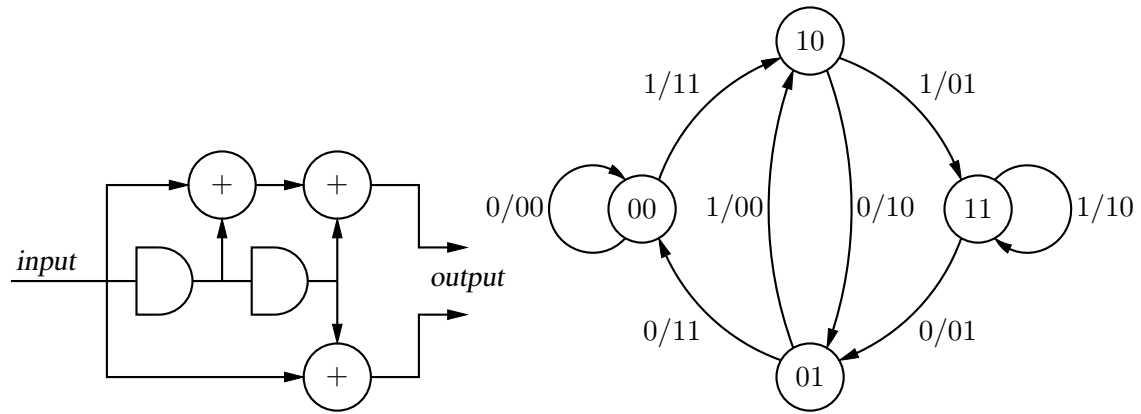


Figure 60: A rate 1/2 binary convolutional coder and its state diagram [ex16.m4].

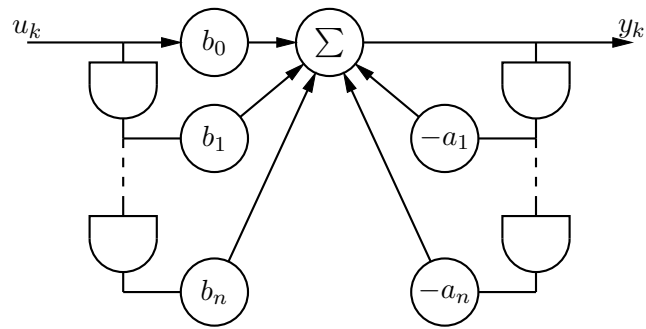


Figure 61: Digital filter [ex03.m4].

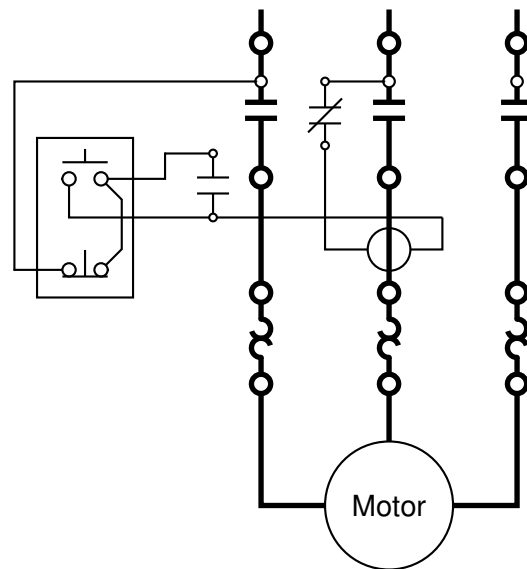


Figure 62: Motor control connections [MotorControl.m4].

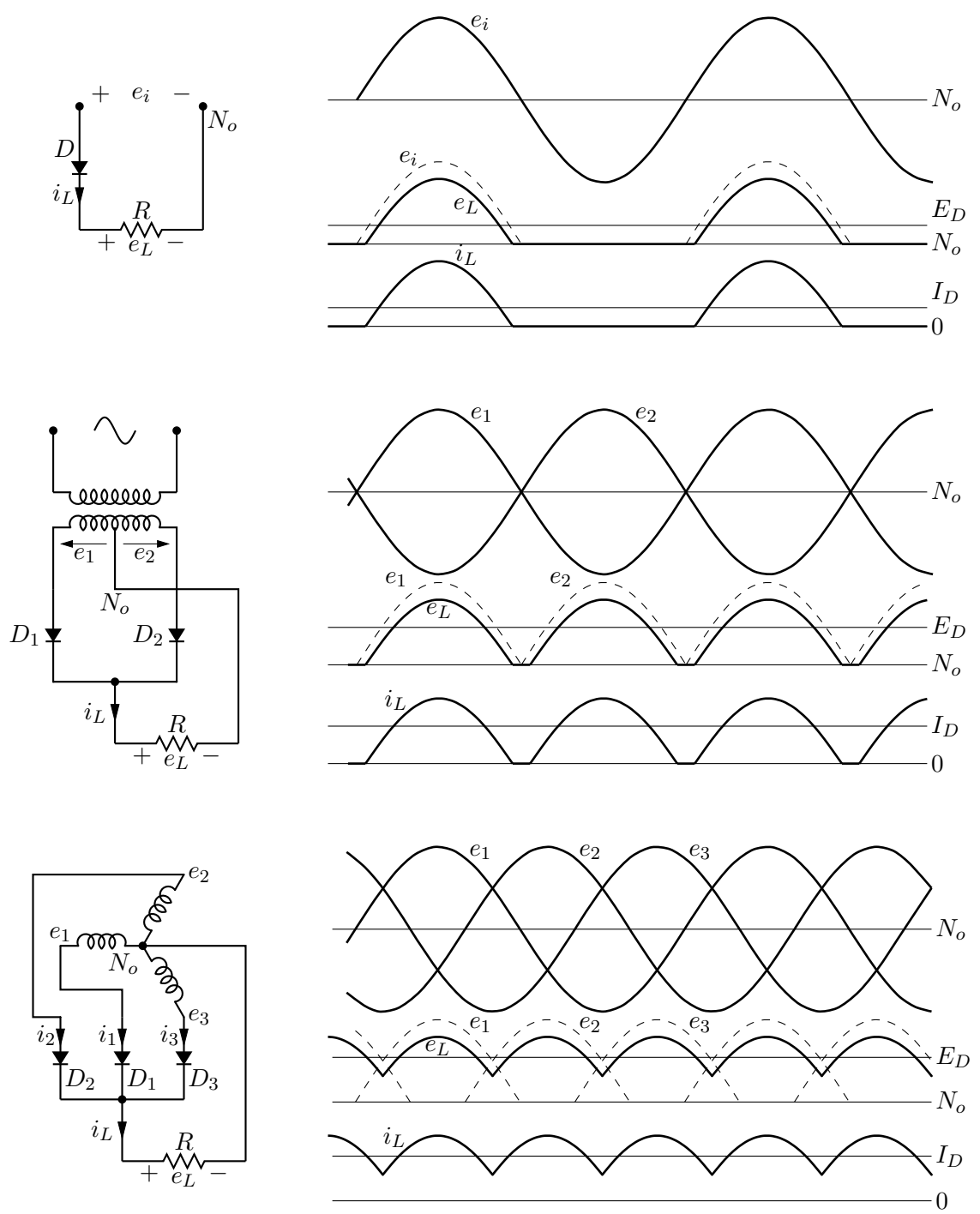


Figure 63: Rectifier circuits and waveforms [Rectifiers.m4].

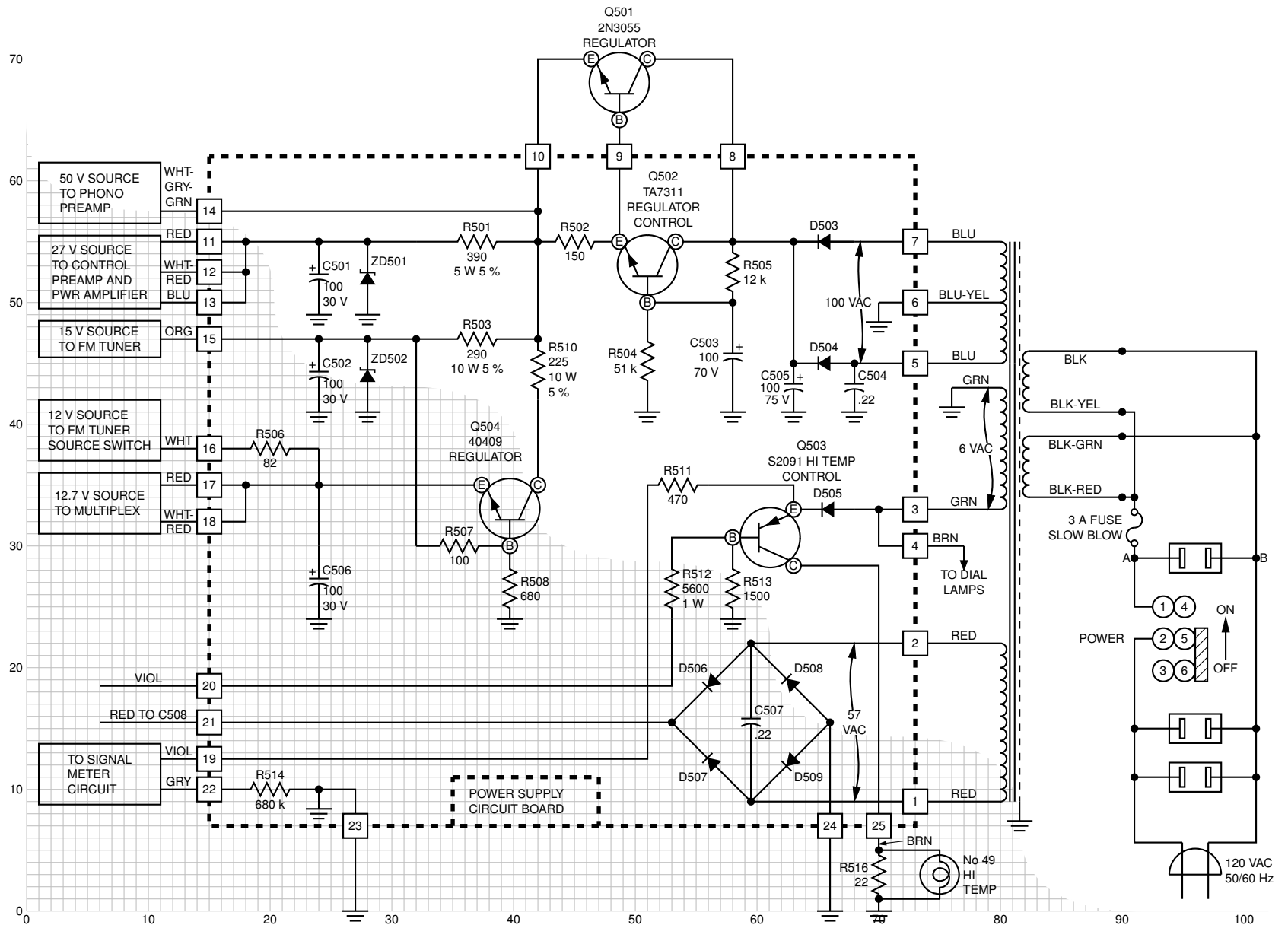


Figure 64: The power supply of a Heathkit AR-15 (Now, *that* was a receiver!) with custom transformer and other elements, drawn on a grid (partially shown) to aid in placement [Heathkit.m4].

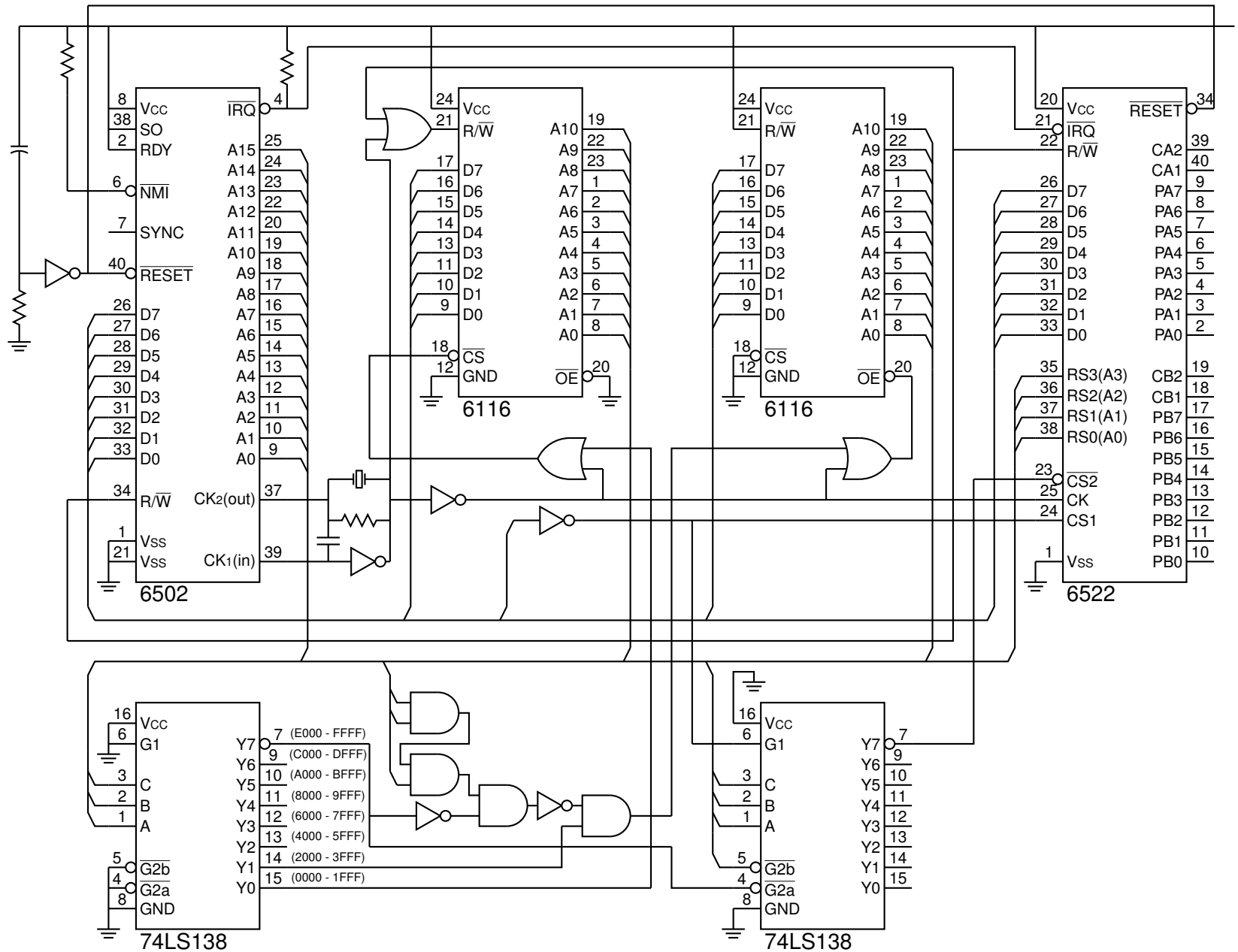


Figure 65: A digital circuit of moderate size, redrawn from M. P. Maclenan and G. M. Burns, "An Approach to Drawing Circuit Diagrams for Text Books," Tugboat (12)1, March 1991, pp. 66-69 [lcct.m4].



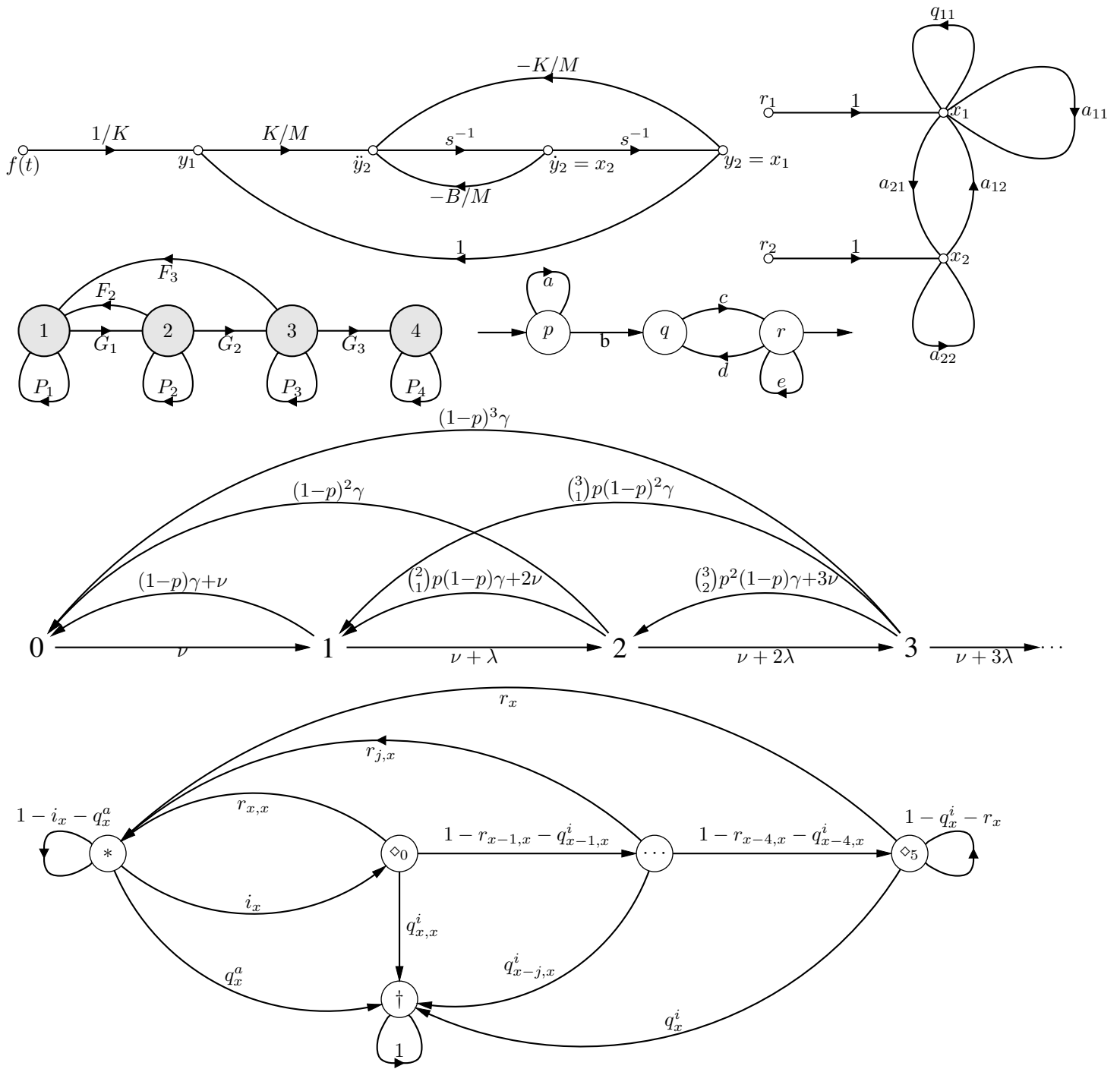


Figure 67: Signal-flow graphs [sfg.m4].



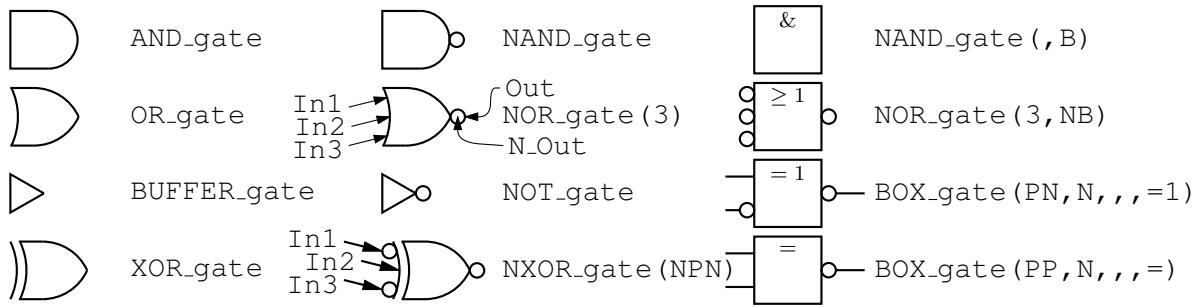


Figure 68: Basic logic gates [Logic.m4].

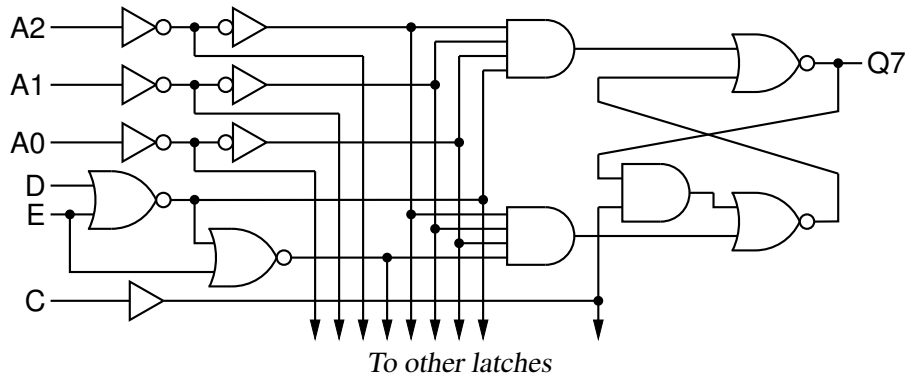


Figure 69: General-purpose latch: a small logic circuit [ex08.m4].

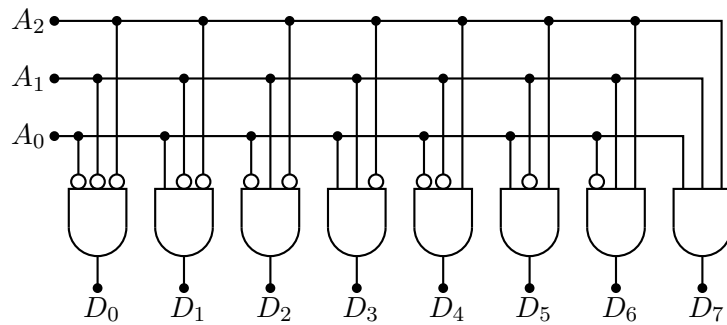


Figure 70: Decoder logic, constructed using the for\_macro [Decoder.m4].

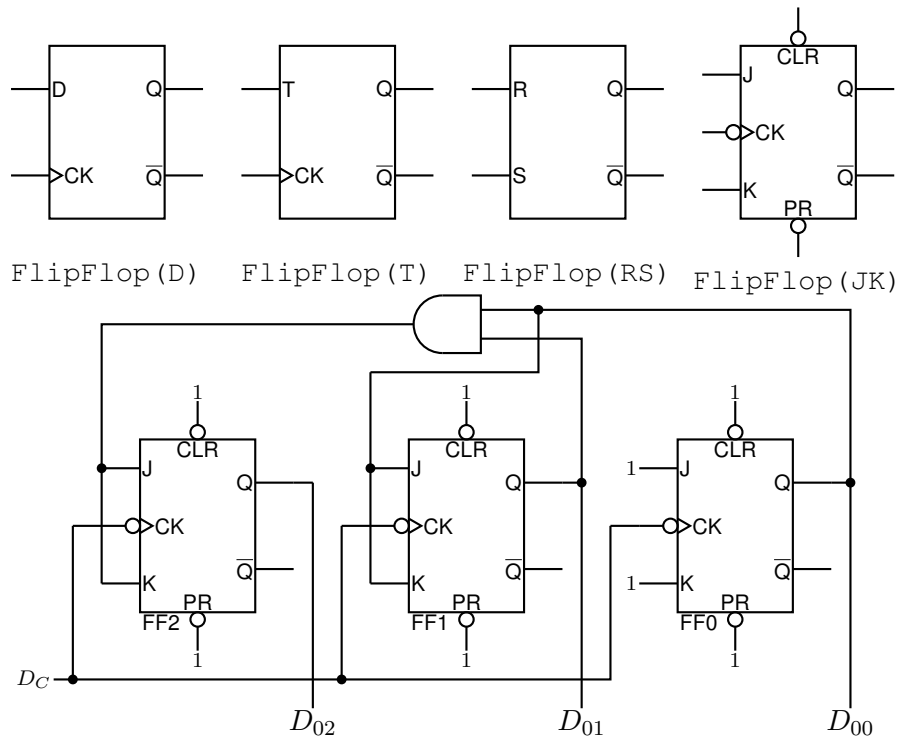


Figure 71: Some flip-flops [ex21.m4].

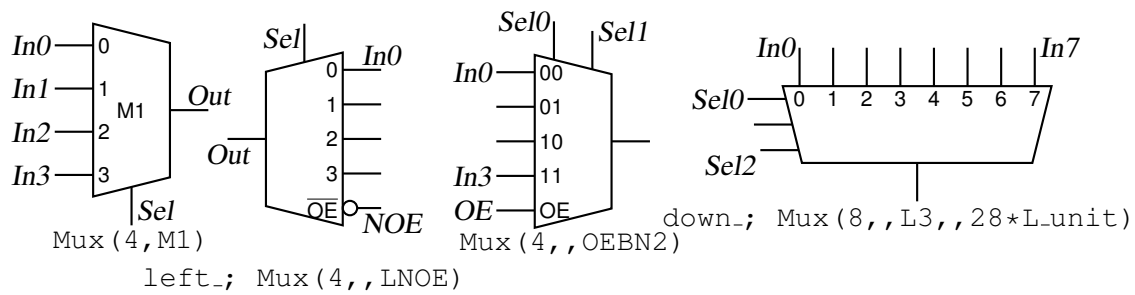


Figure 72: Multiplexer [Multiplexer.m4].

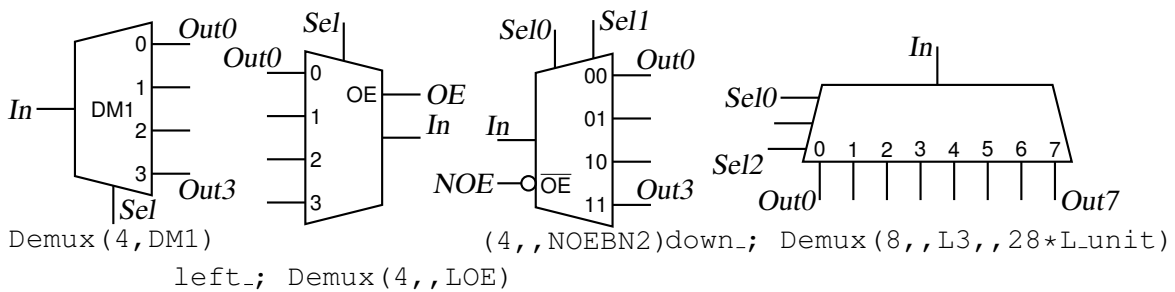


Figure 73: Demultiplexer [Demultiplexer.m4].

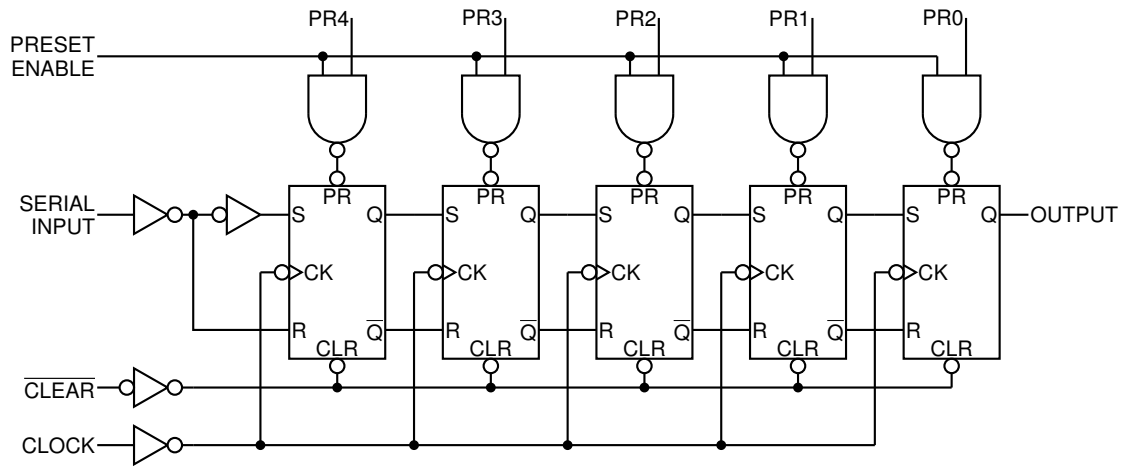


Figure 74: A 5-bit shift register drawn using a custom flip-flop [ShiftR.m4].

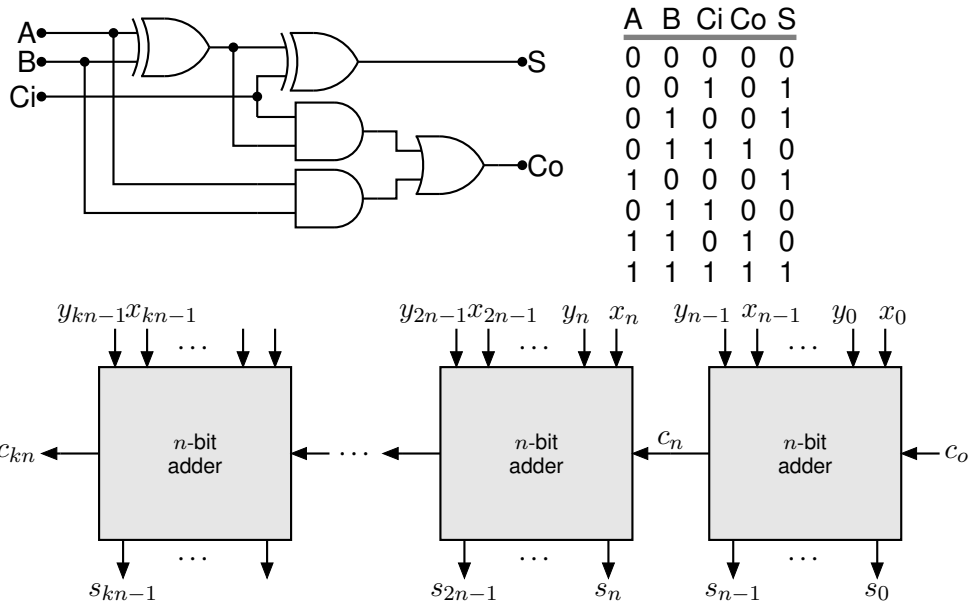
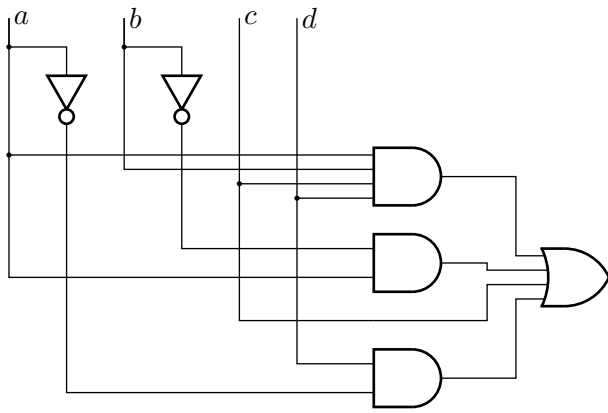
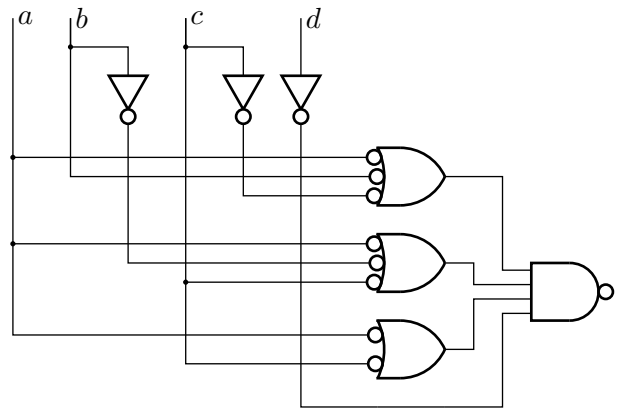


Figure 75: A full adder and a cascade of  $n$ -bit adders [Adder.m4].



CanLogic(AND, OR, abcd, ~ba, c, d~a)



CanLogic(OR, N, NAND, ab~c, a~bc, ac, ~d)

Figure 76: A way of automatically drawing two-layer logic diagrams [CanLogic.m4].

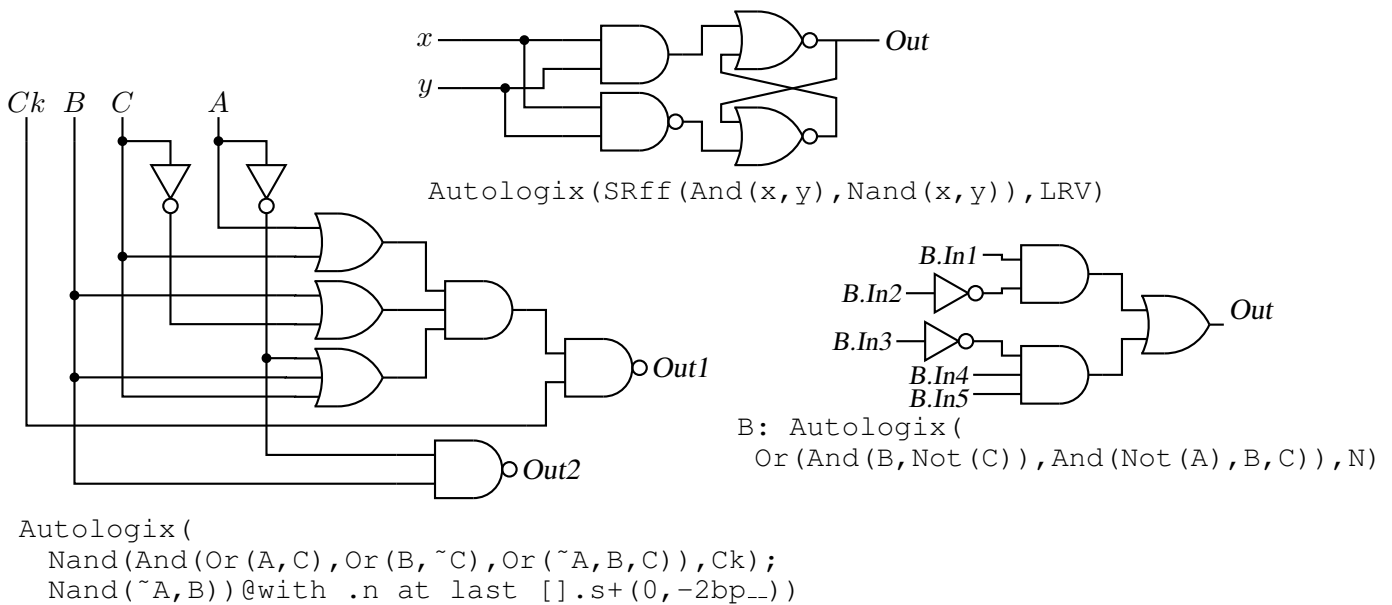
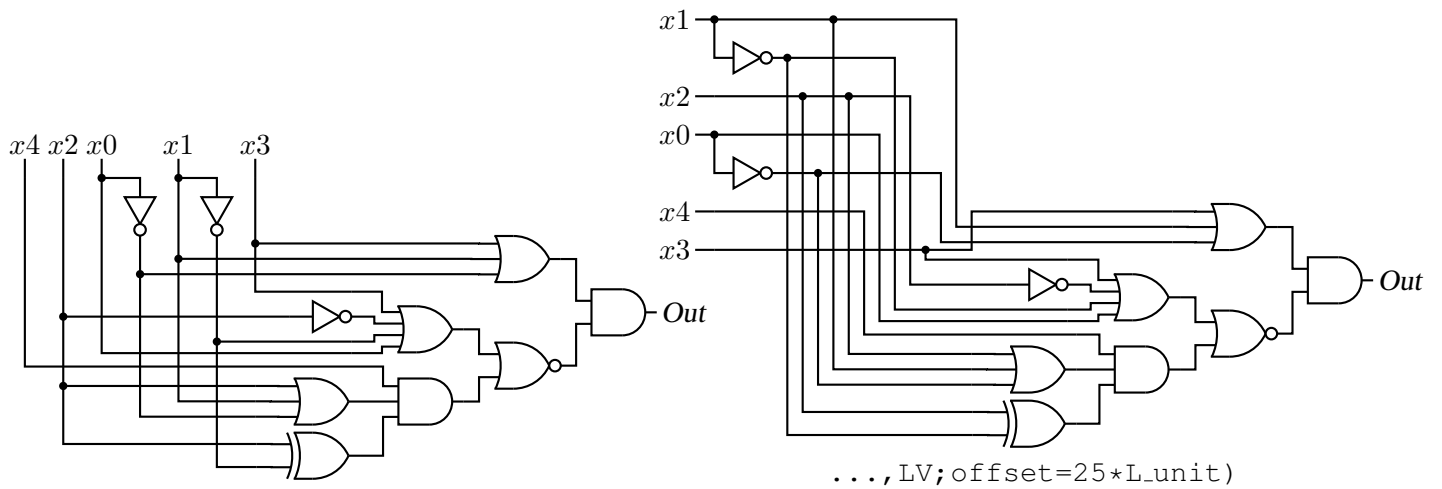


Figure 77: The Autologix (Boolean expression; Boolean expression... , options) macro automatically draws Boolean expressions in function notation. The function tree is drawn, then a row or column of inputs, then the connections. The default result is on the left, a custom element at the top, and a tree of gates only is shown on the right. [Alogix.m4].



```
Autologix (And (Or (x3, x1, ~x0),
  Nor (Or (x3, Not (x2), ~x1, x0), And (x4, Or (x2, x1, ~x0), Xor (x2, ~x1))))))
```

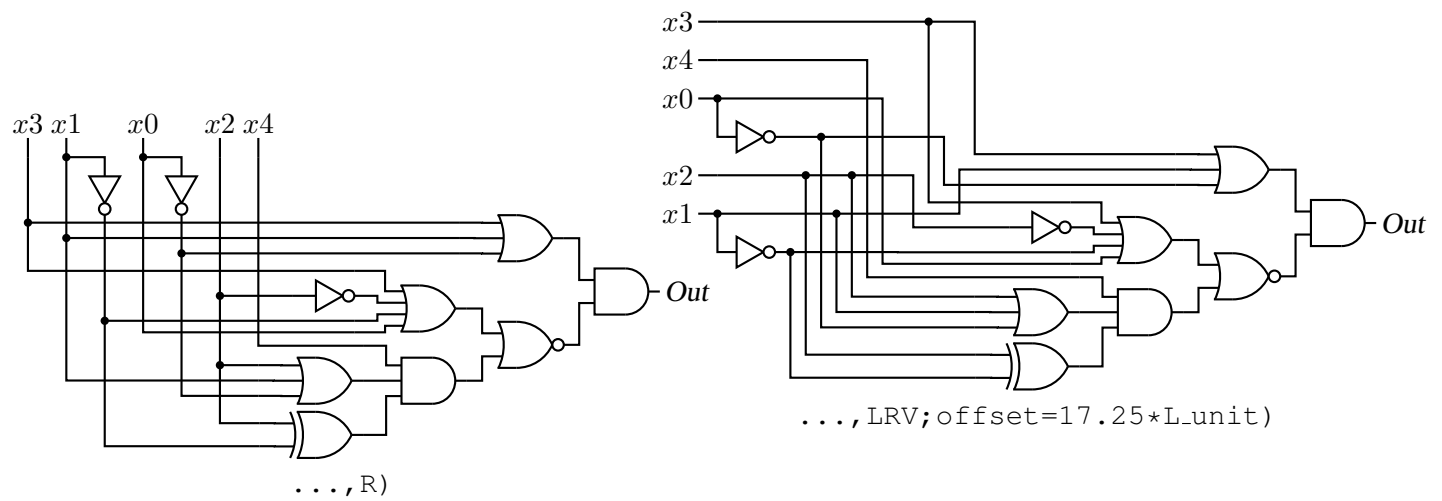


Figure 78: The Autologix macro can draw inputs on the left but the added drawing complexity may require hand tuning with second-argument options: L puts the inputs on the left, R reverses their order, V scans the input arguments in reverse order, and `offset=value` displaces the array of inputs [ABlogix.m4].

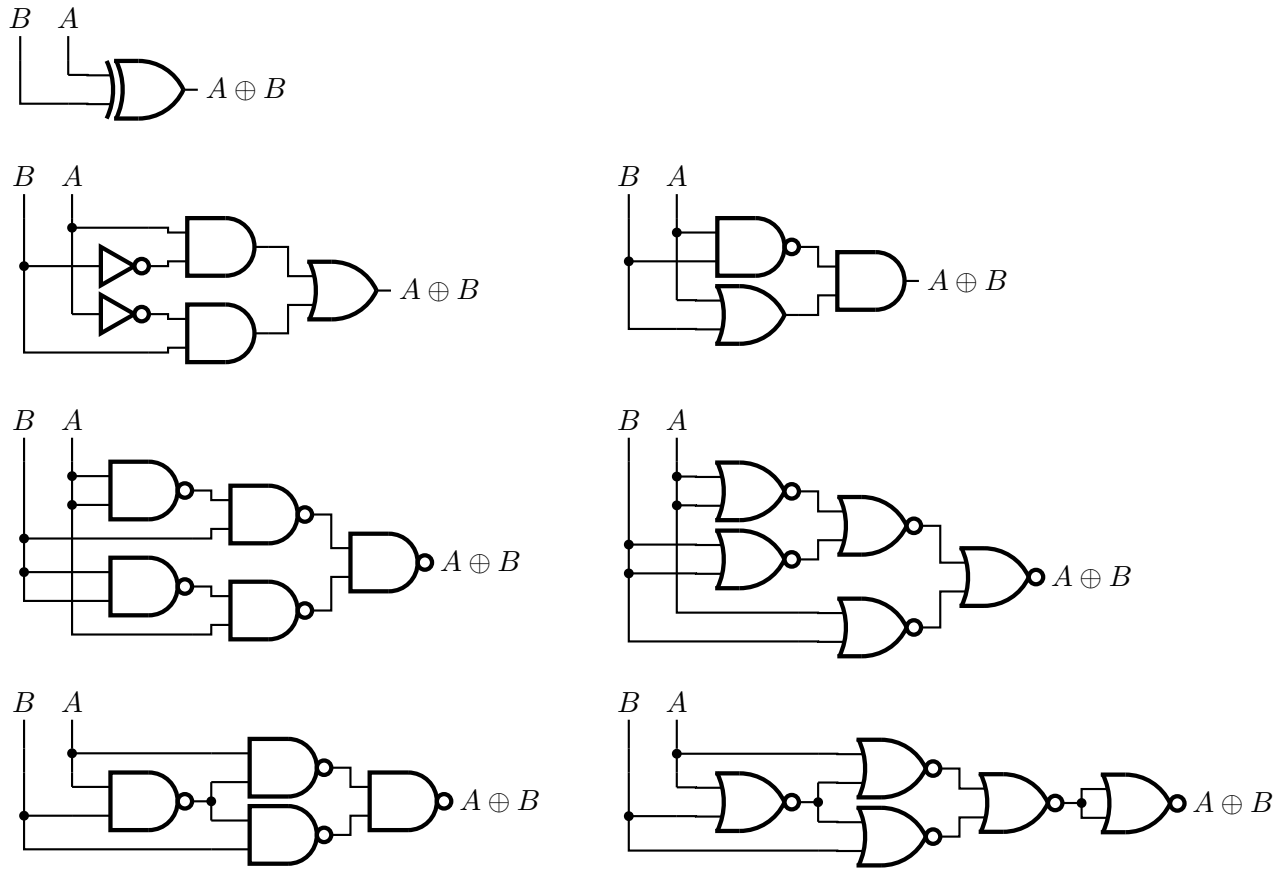


Figure 79: Realizations of the XOR function using Autologix [XOR.m4].

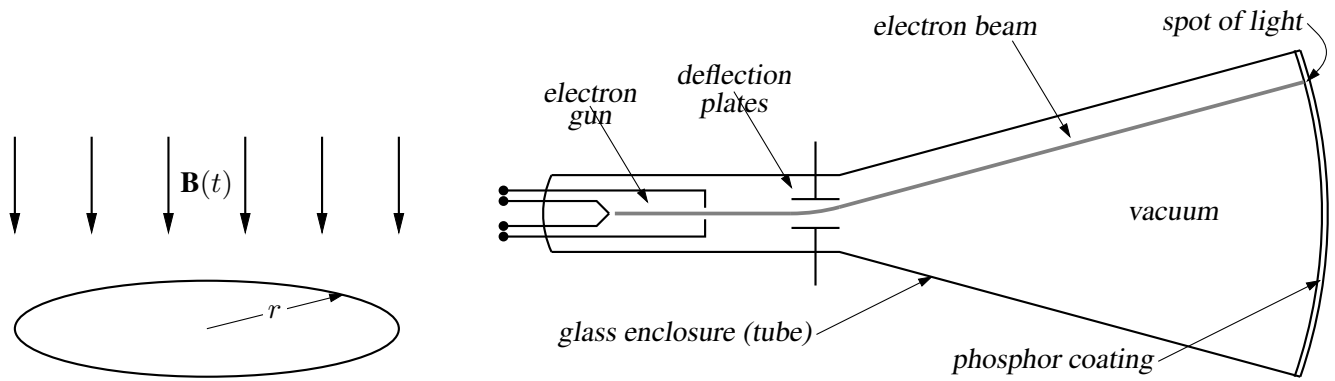


Figure 80: Line diagrams [ex00.m4].

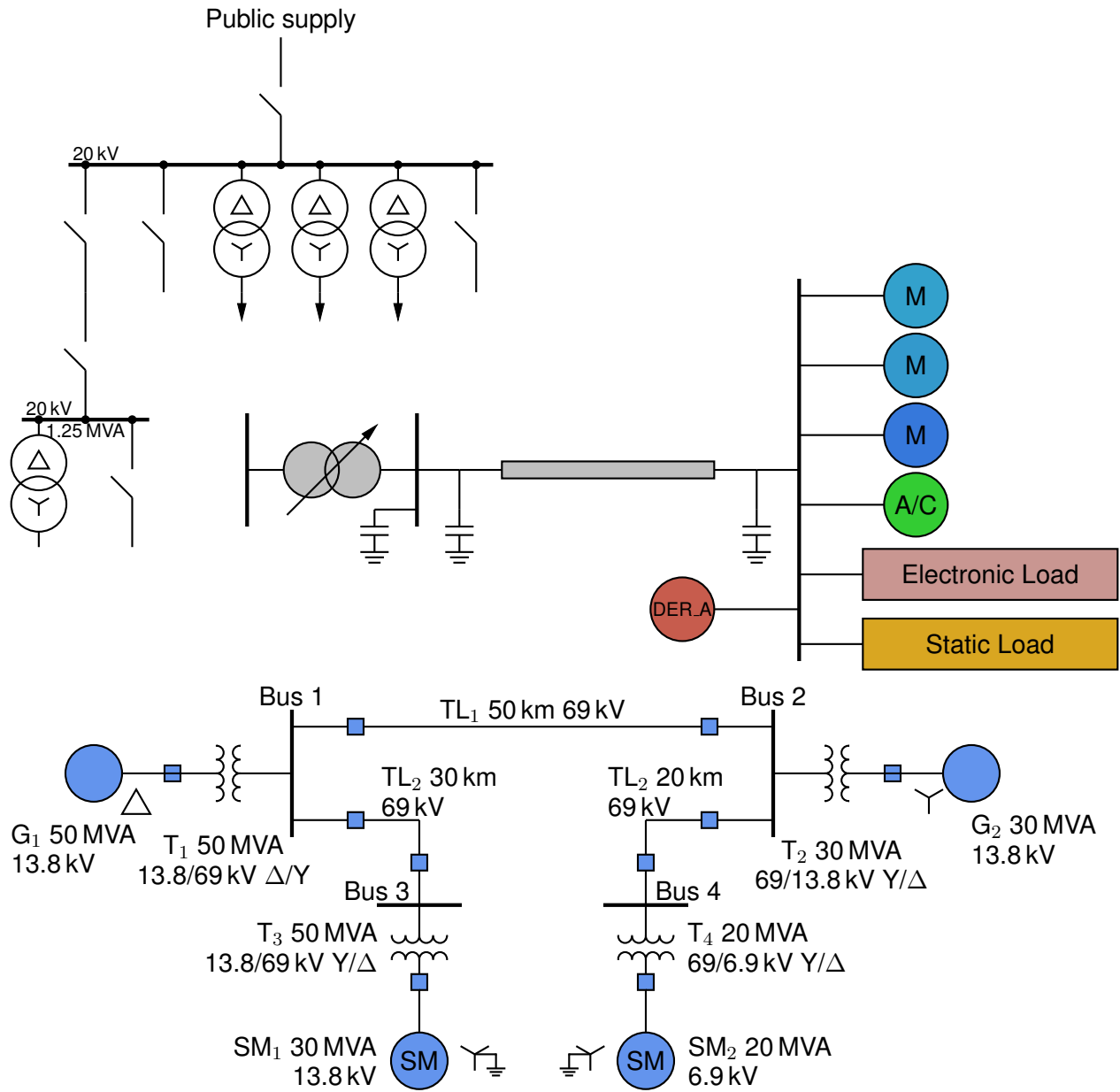
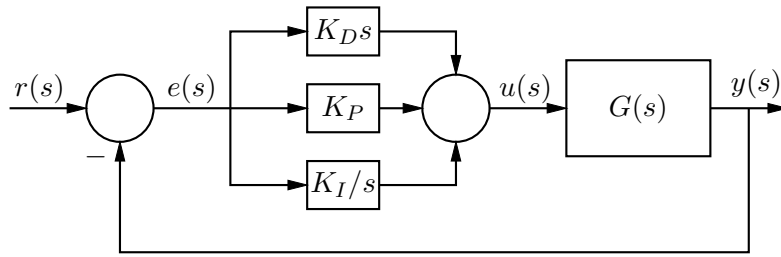
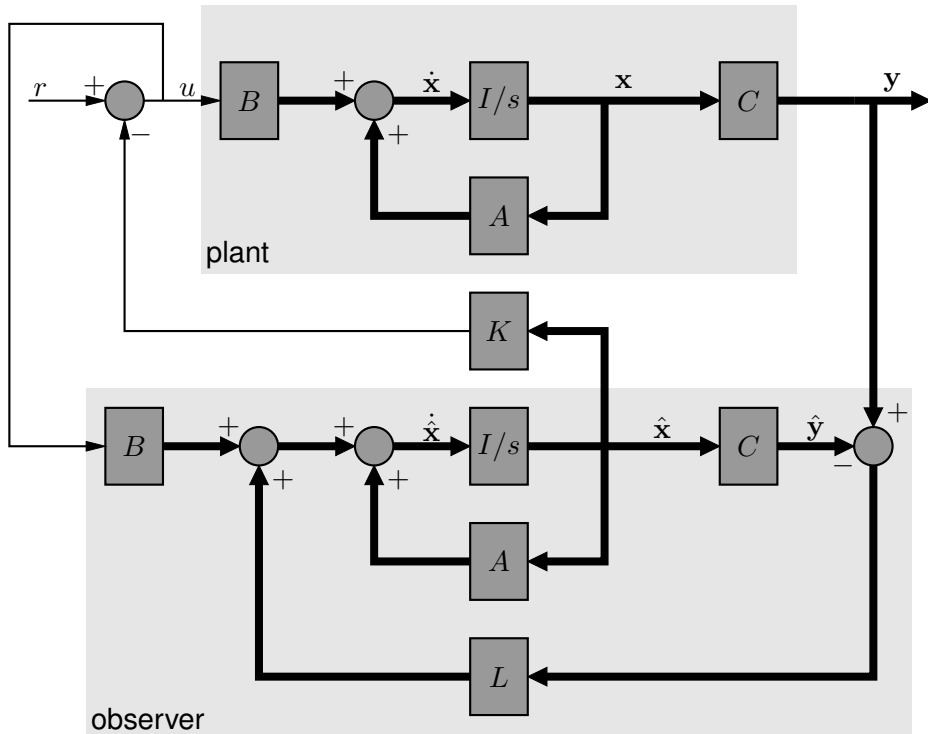


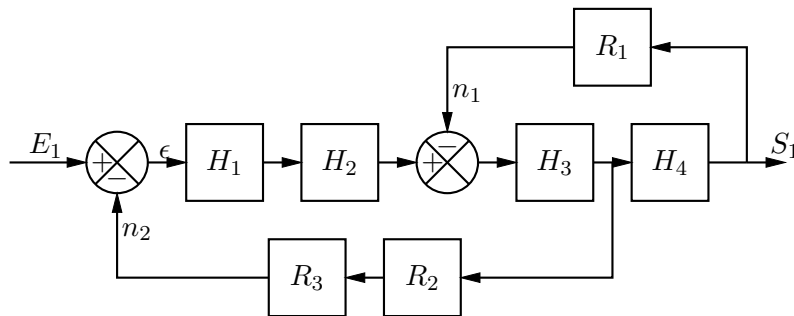
Figure 81: A test of experimental single-line diagram macros [EEP.m4].



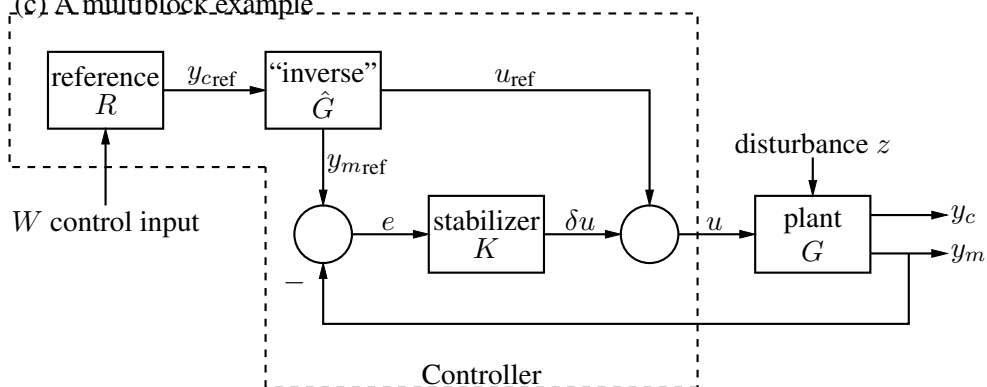
(a) *PID* control



(b) Single-input plant with feedback from a full-order observer



(c) A multiblock example



(d) Nonlinear feedforward (for performance) and small-signal feedback (for stability)



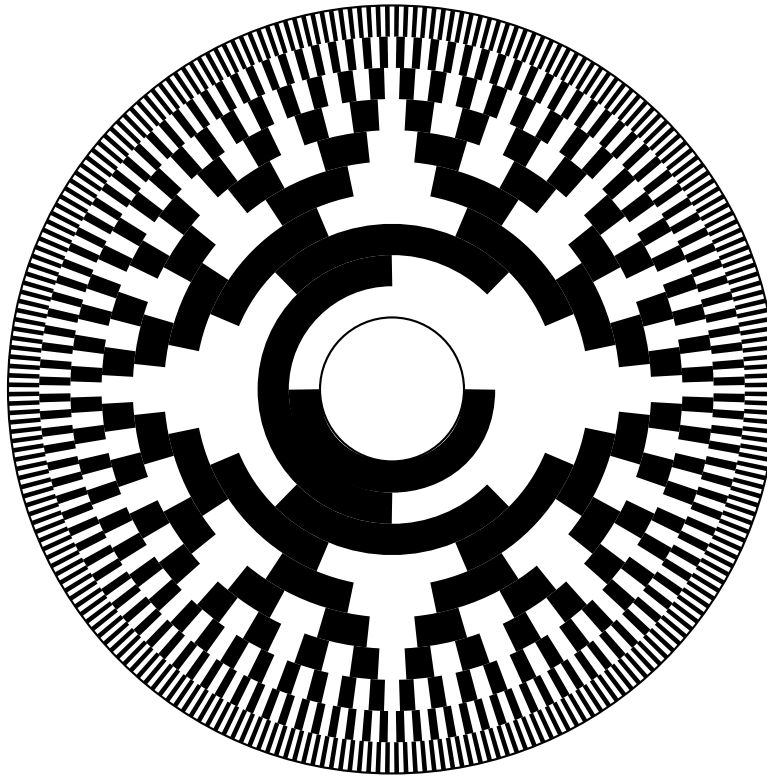


Figure 83: Gray code 10-bit encoder disk pattern [GrayCode.m4].

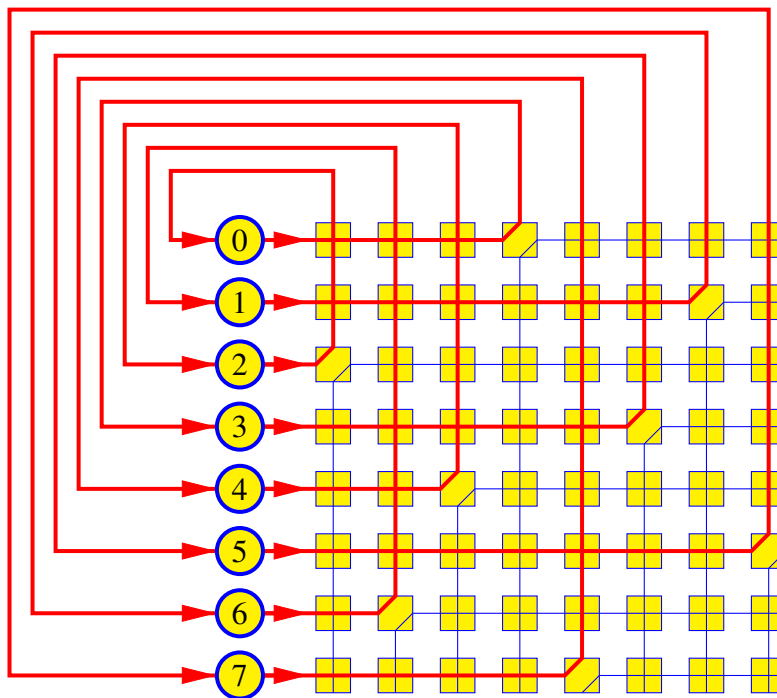


Figure 84: A crossbar switch [Crossbar.m4].

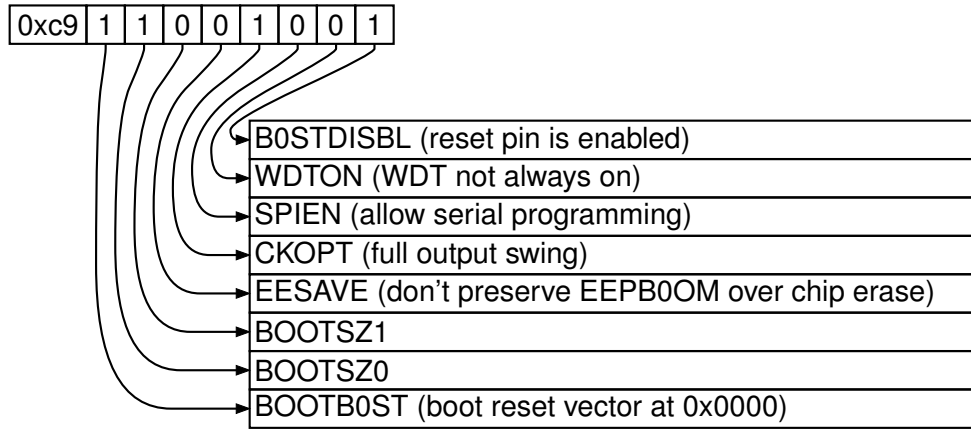


Figure 85: Elementary splines [Byte.m4].

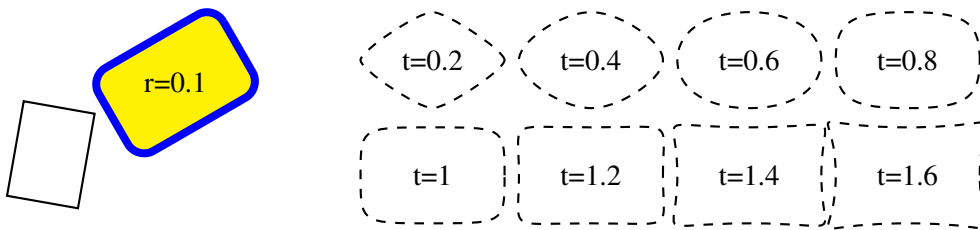


Figure 86: The macro `rotbox(wid,ht,type,[r|t=val])` draws a box in the current direction [Rotbox.m4].

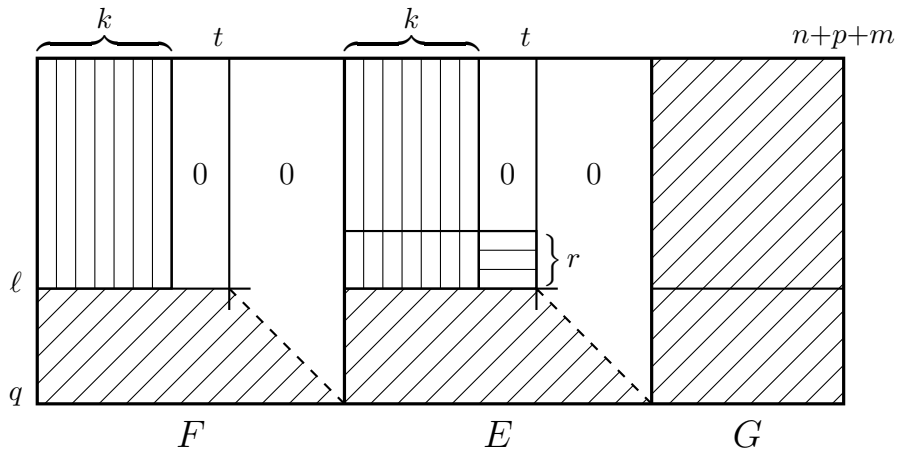
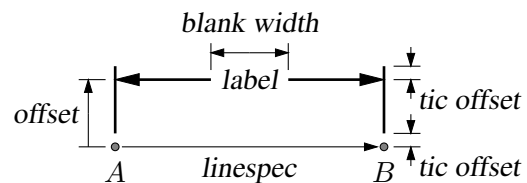
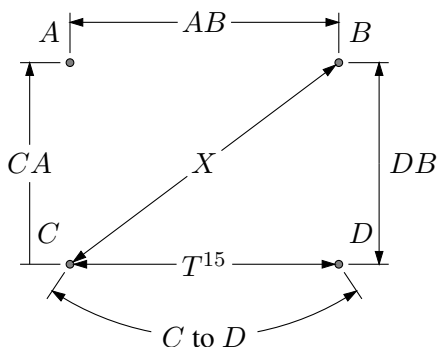
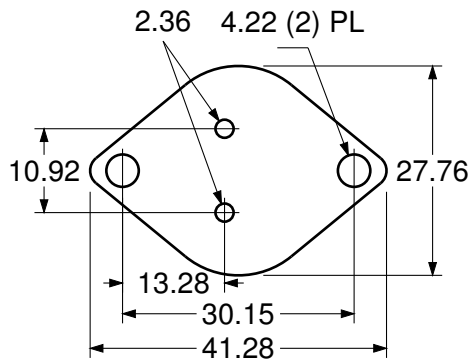


Figure 87: Crosshatching [ex06.m4].



```

dimension_(from A to B,0.3,$AB$,20bp_)
dimension_(from C to B,,,$X$,16bp_)
dimension_(from C to A,0.3,$CA$,14bp_,-,->)
dimension_(from D to B,-0.3,"$DB$" ljust)
dimension_(from C to D,,s_box($T^{%g}$,15),W)
arcdimension_(from C to D with .c at \
0.5 between A and B,12bp_-,s_box($C$ to $D$),W)

```

```
dimension_(from A to B,0.5,\sl label,29bp_-,0.1)
```

Figure 88: Illustrating the macro `dimension_(linespec, offset, label, D|H|W|blank width, tic offset,<-|->)`. A negative second argument implies an offset to the right of the `linespec` direction. A `label` starting with `"` or `sprintf` is copied literally. If `label` is an `s_box(...)` then setting argument 4 to `H`, `W`, or `D` tailors the blank width to the `s_box` height, width, or diagonal respectively; i.e., `W` is equivalent to `s_wd+textoffset*2`. The macro `arcdimension_` is similar but the first argument specifies the arc to be dimensioned and the second argument is the outward radial offset of the dimension arrow arc. [ex09.m4].

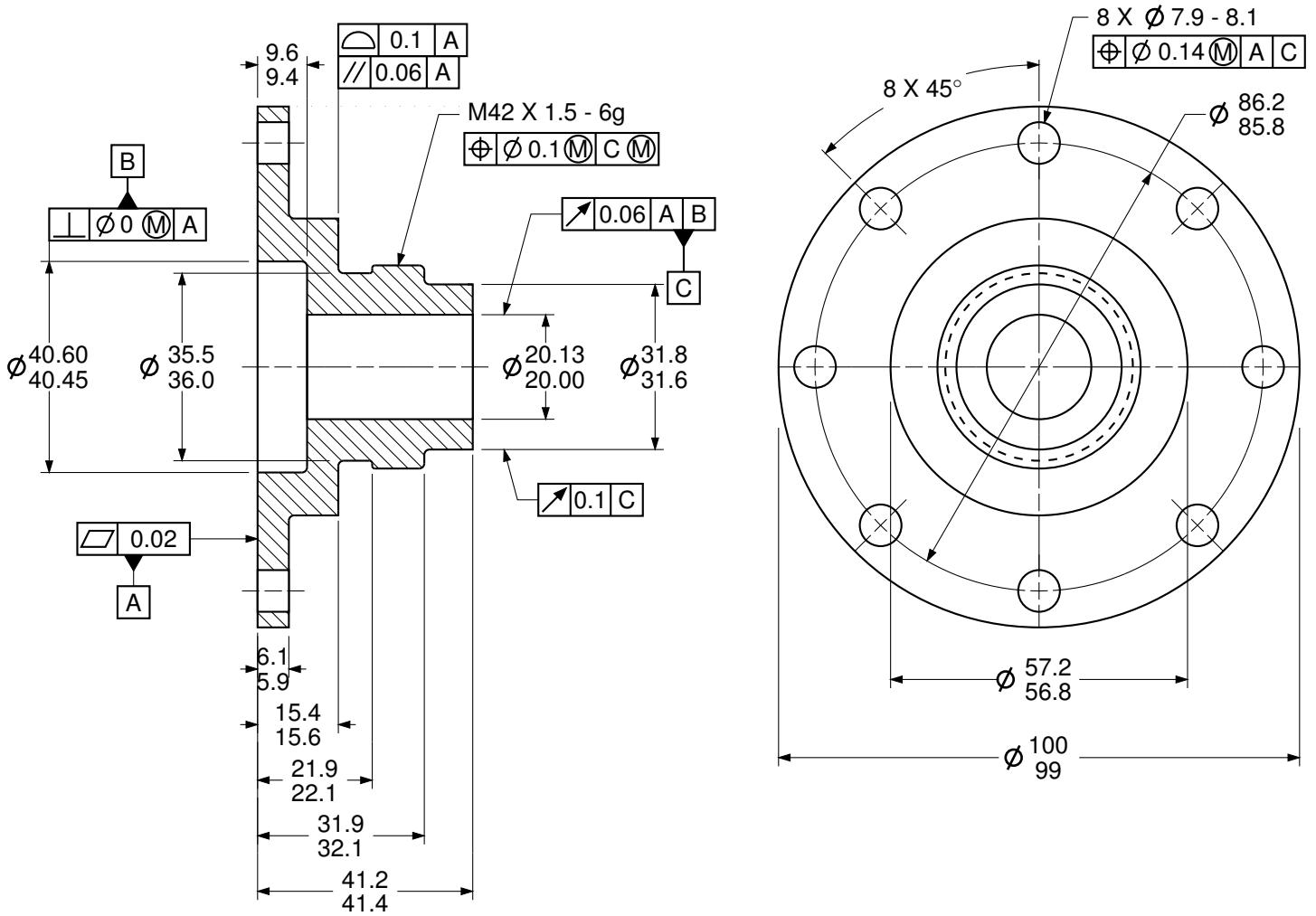


Figure 89: Dimensioning with tolerances [Plate.m4].

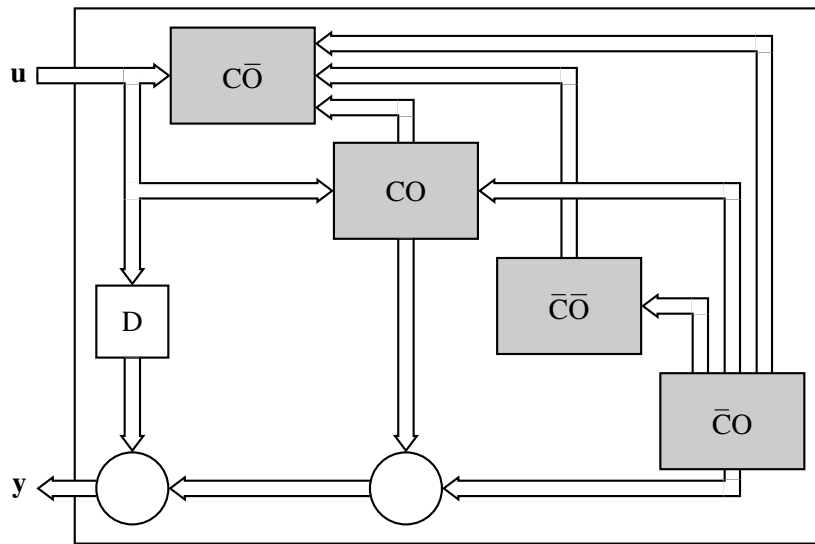


Figure 90: The Kalman canonical decomposition of a dynamical system: use of darrow [ex05.m4].

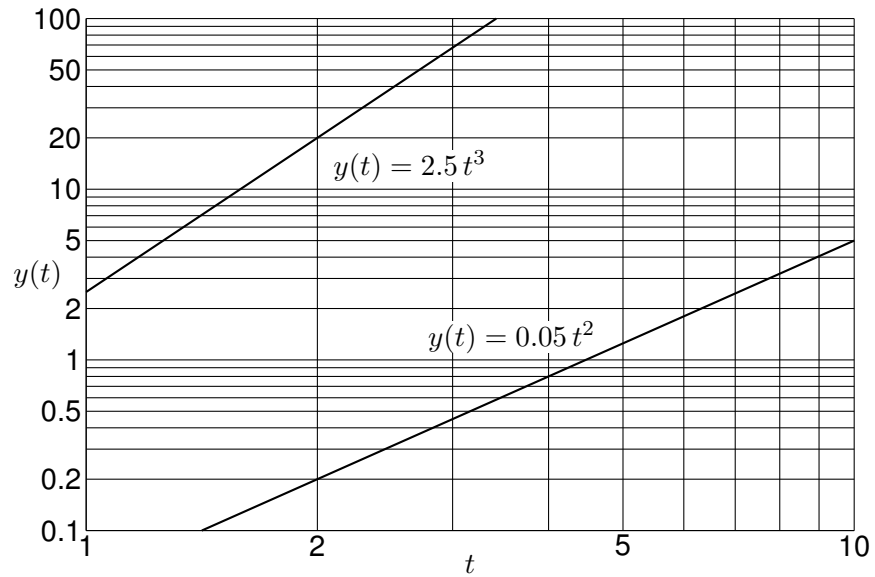


Figure 91: A graph drawn using the pic language [Loglog.m4].

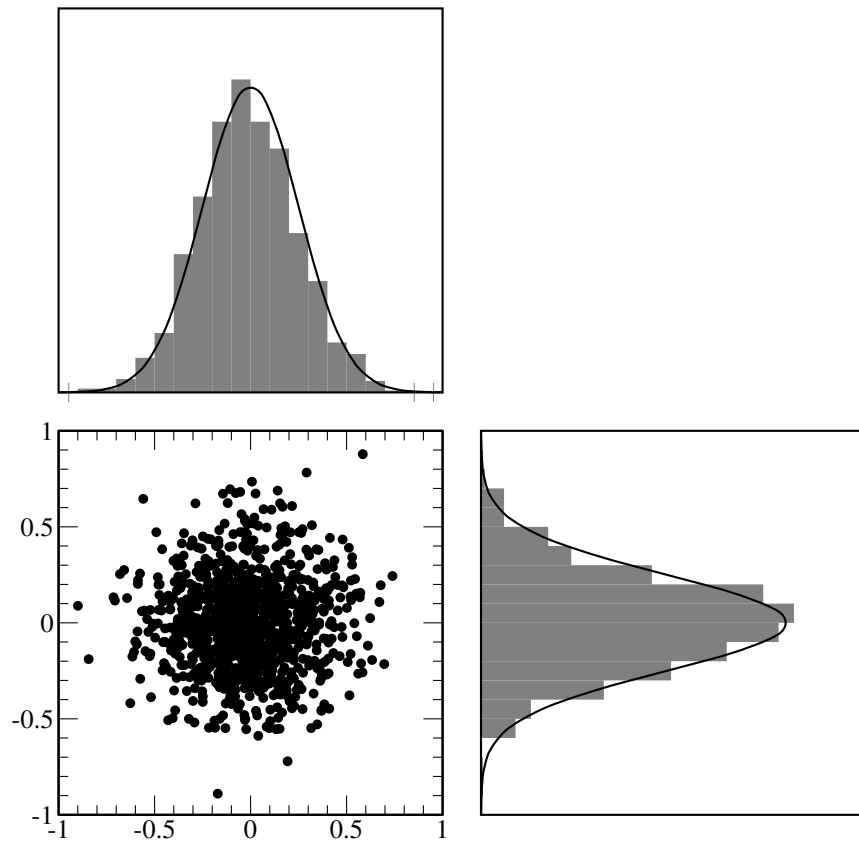


Figure 92: Testing random number generation using dpic macro `randn (array name, n, mean, std dev)` which calls pic built-in `rand()` [random.m4].

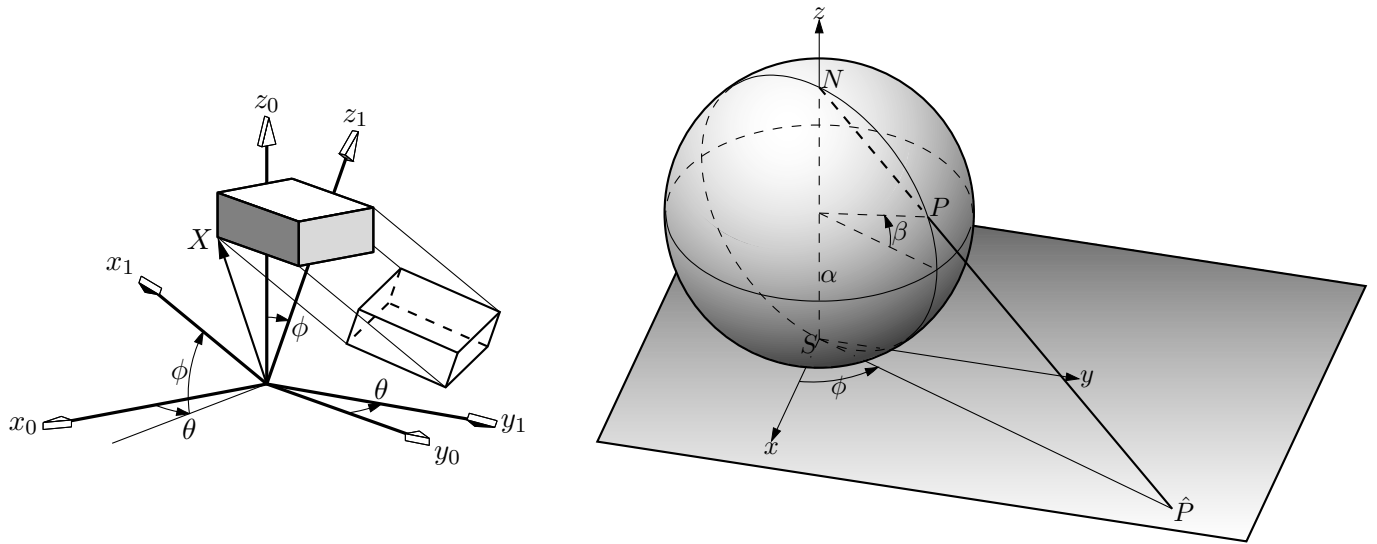


Figure 93: Test of `project` and other `lib3D` macros, showing the projection of a solid onto the  $y_1, z_1$  plane by sighting along the  $x_1$  axis. [`exp.m4`].

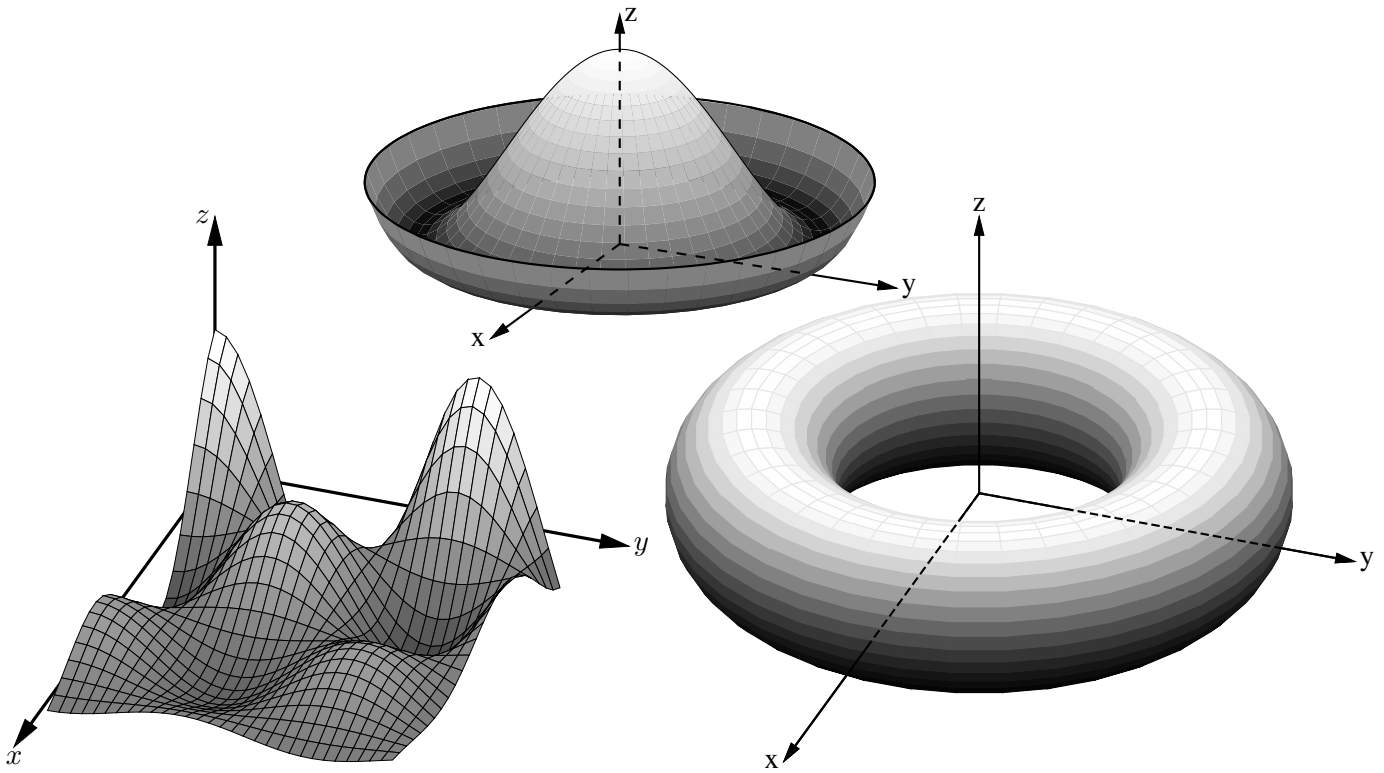


Figure 94: Plotting surfaces using gray scales [`graysurf.m4`].

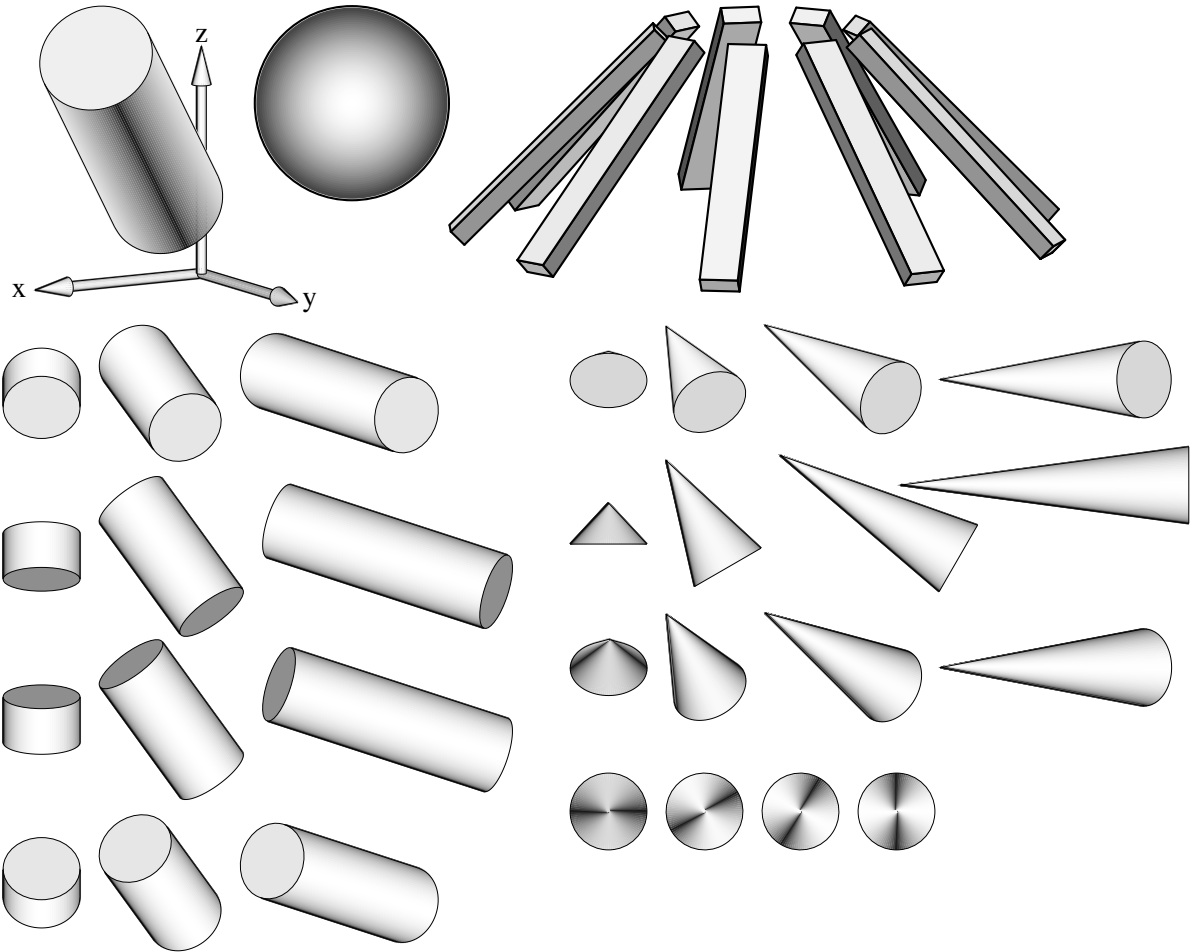


Figure 95: Basic shapes [shapes.m4].

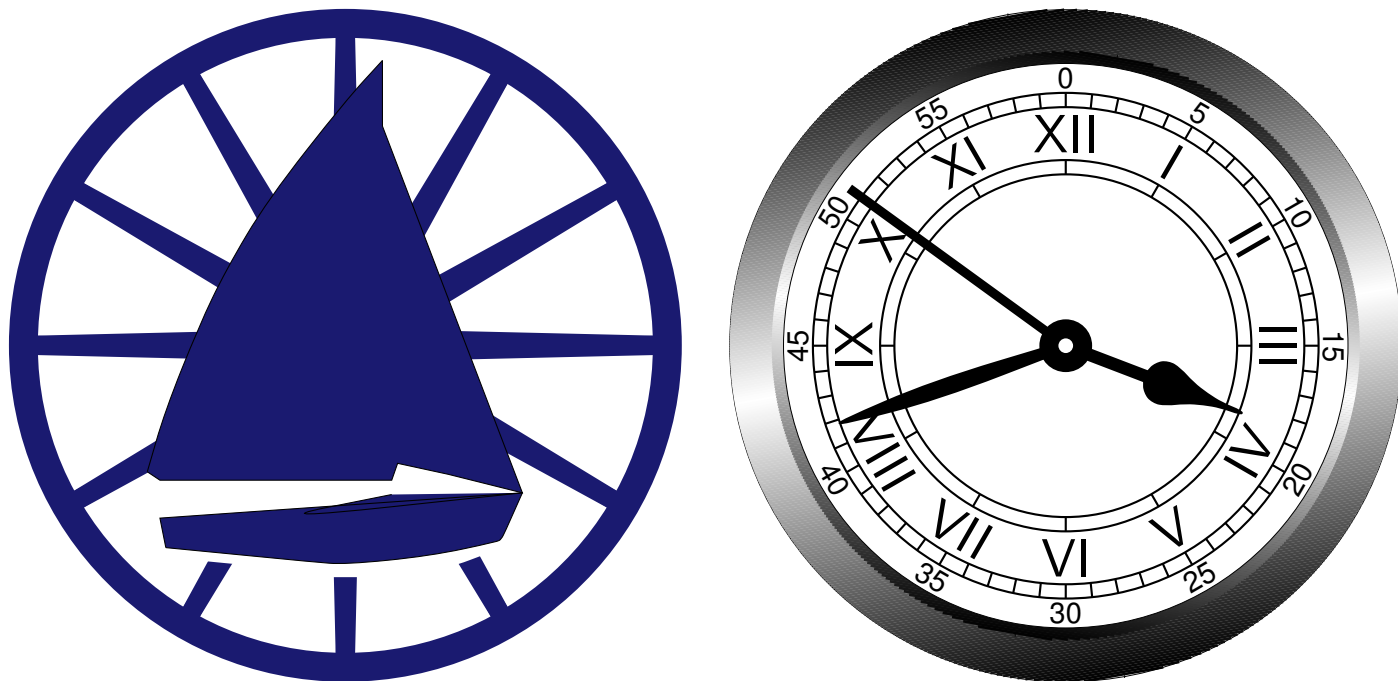


Figure 96: Conestoga Sailing Club (illustrating the filling of arbitrary shapes) and an antique clock face with shading and rotated text [csc.m4].

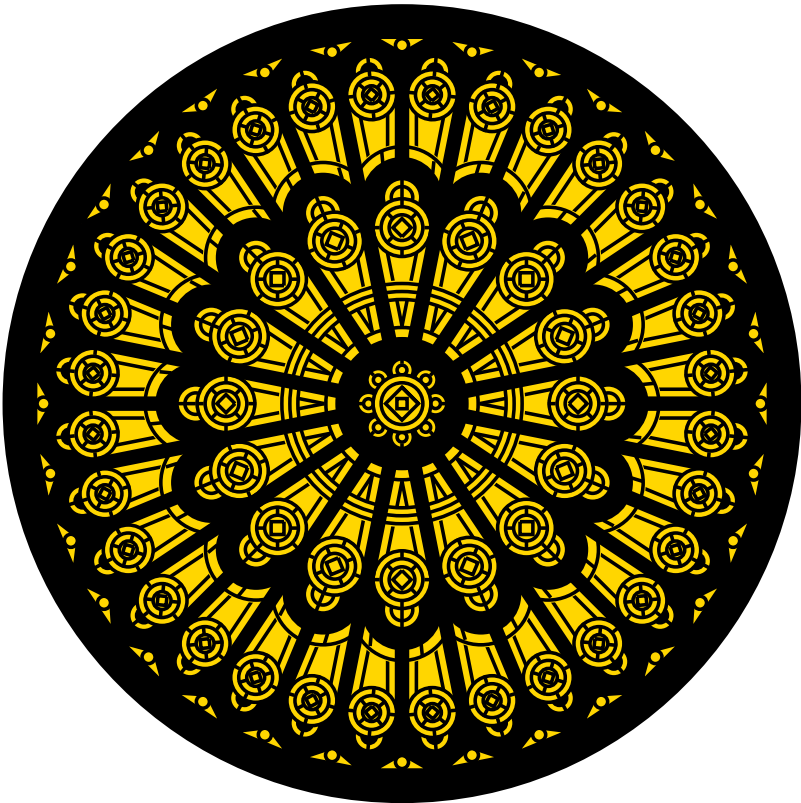


Figure 97: Redrawn from a detail of the set design for the musical *Dracula*, used for testing `dpic`. This diagram consumes much  $\text{\LaTeX}$  main memory but can be produced directly as pdf using `dpic -d`, as svg using `dpic -v`, or as postscript using `dpic -r` since no text formatting is required [rose.m4].



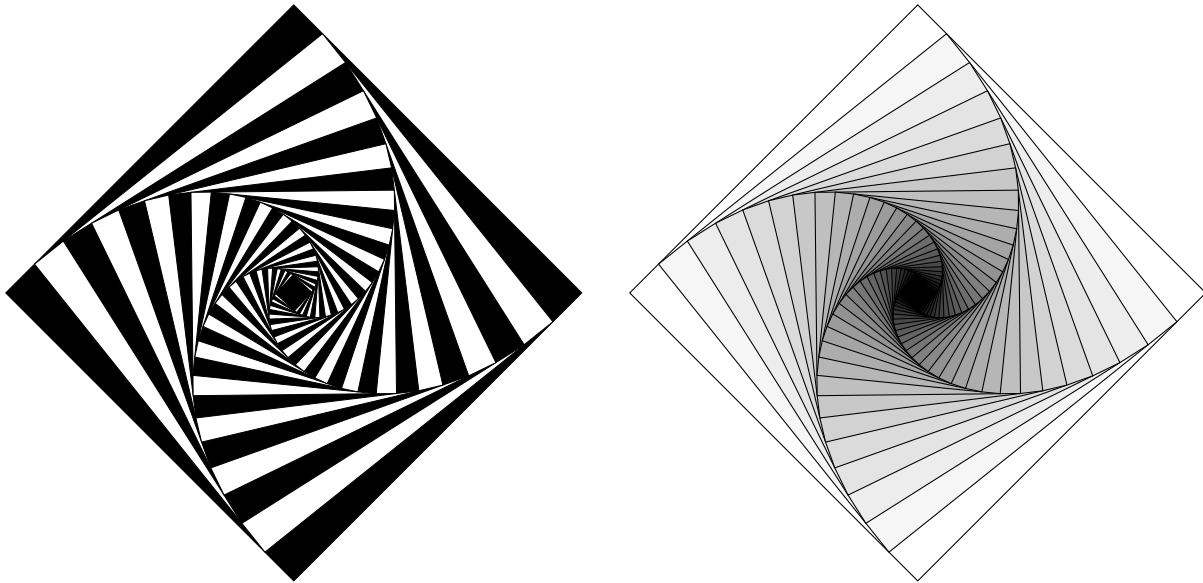


Figure 98: Variations on M. Goossens, S. Rahtz, and F. Mittelbach, *The L<sup>A</sup>T<sub>E</sub>X Graphics Companion*, Addison-Wesley 1997, pp. 57-58 [diamond.m4].

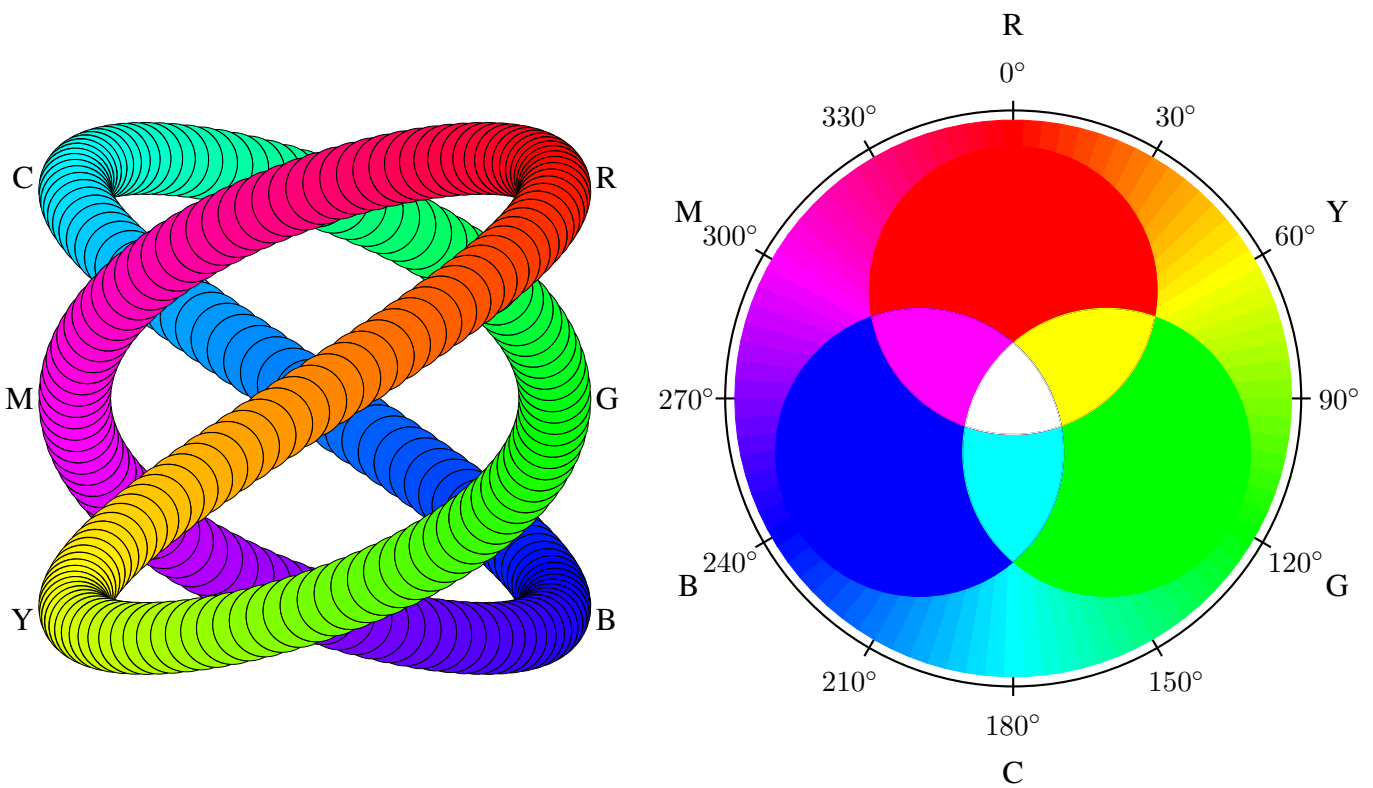


Figure 99: An exercise in calculating RGB colours [worm.m4].

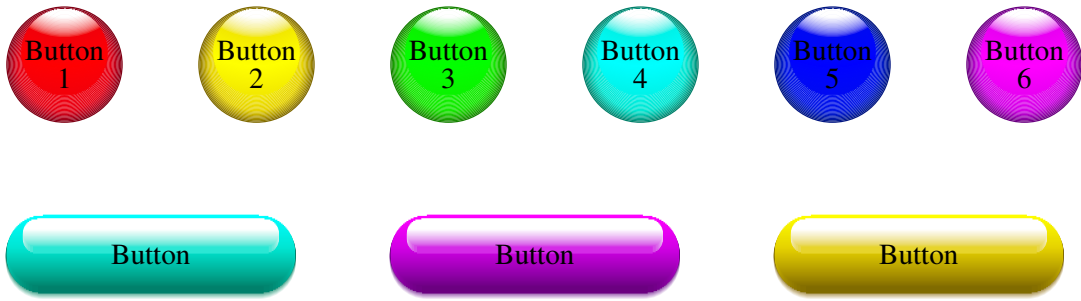


Figure 100: Shading in color [Buttons.m4].

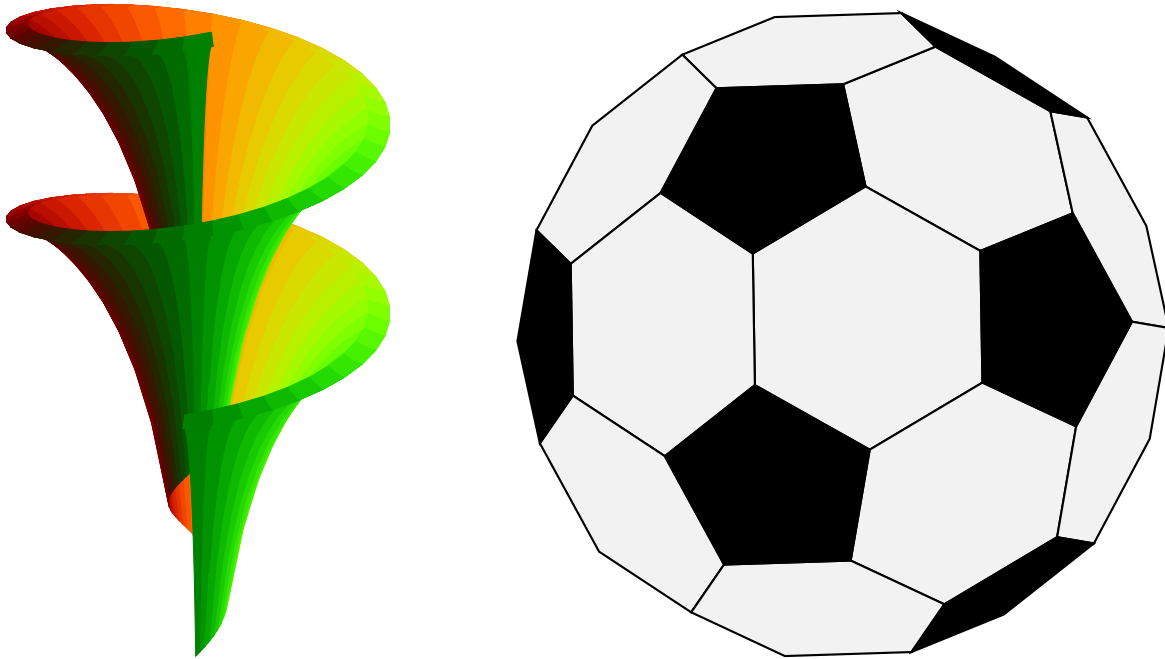


Figure 101: Dini surface and an icosahedron [Dini.m4].

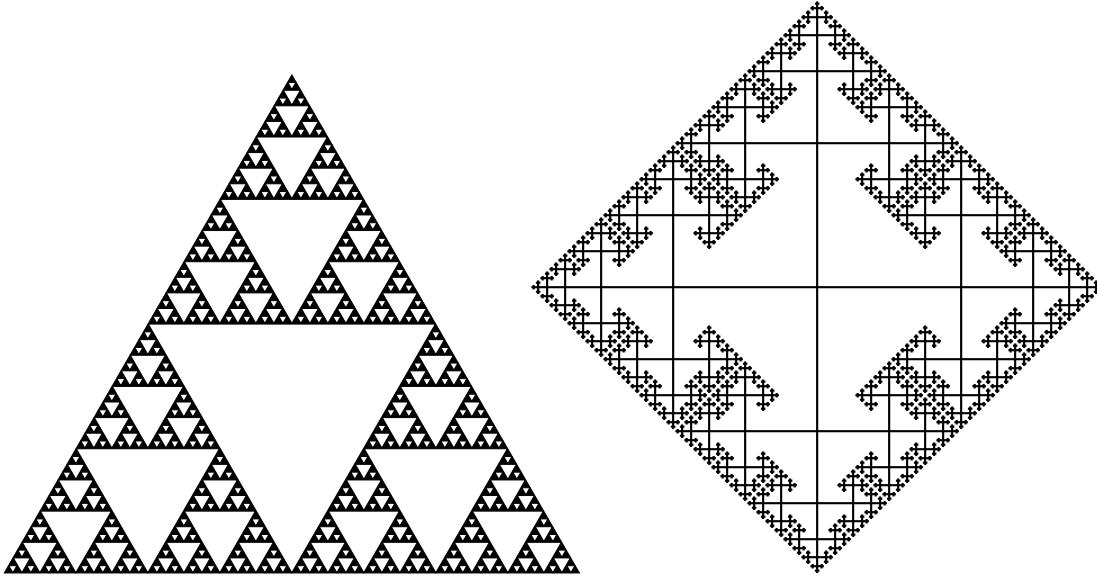


Figure 102: The Sierpinski triangle and a Cayley graph: tests of pic macro recursion [Sierpinski.m4].

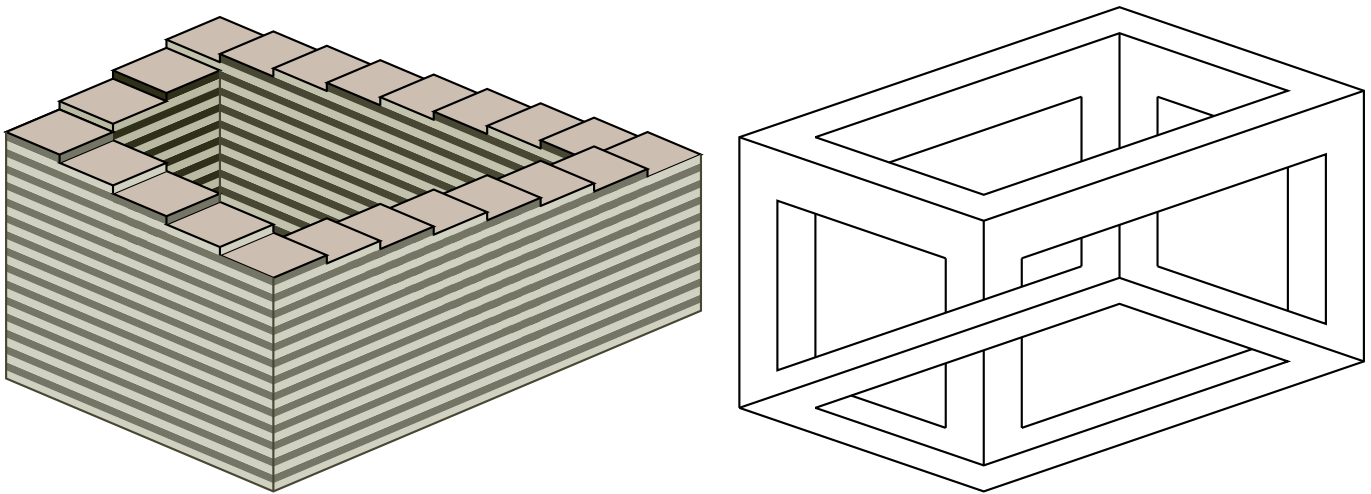


Figure 103: Penrose stairs and an Escher-like object [Escher.m4].

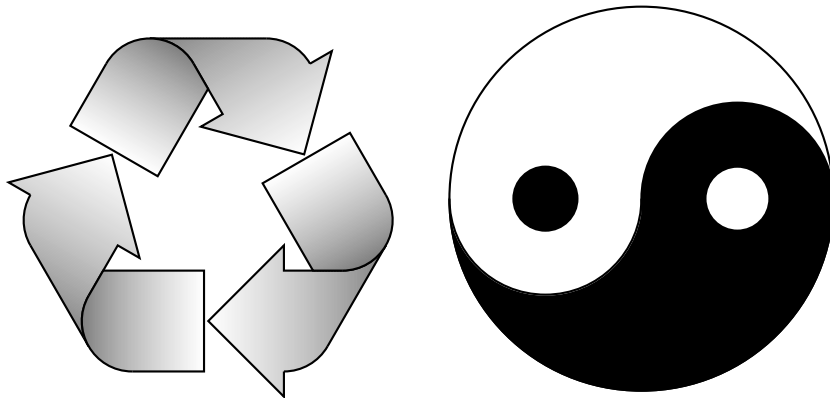


Figure 104: Modest repetition and partial fill [recycle.m4].

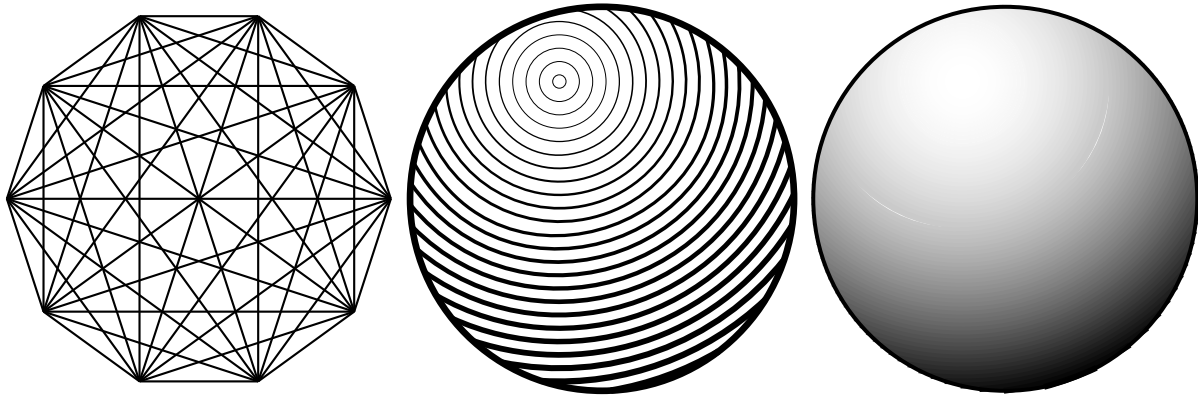


Figure 105: Simple diagrams that are easily drawn by looping [ex15.m4].

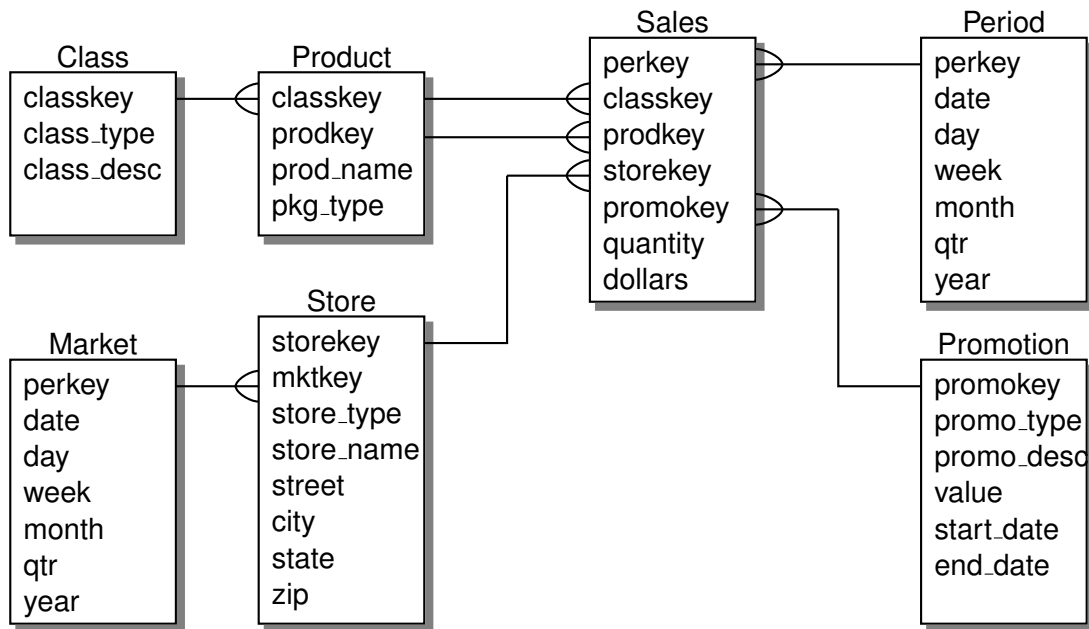


Figure 106: Illustrating shadebox and a custom crowfoot line termination [Crow.m4].

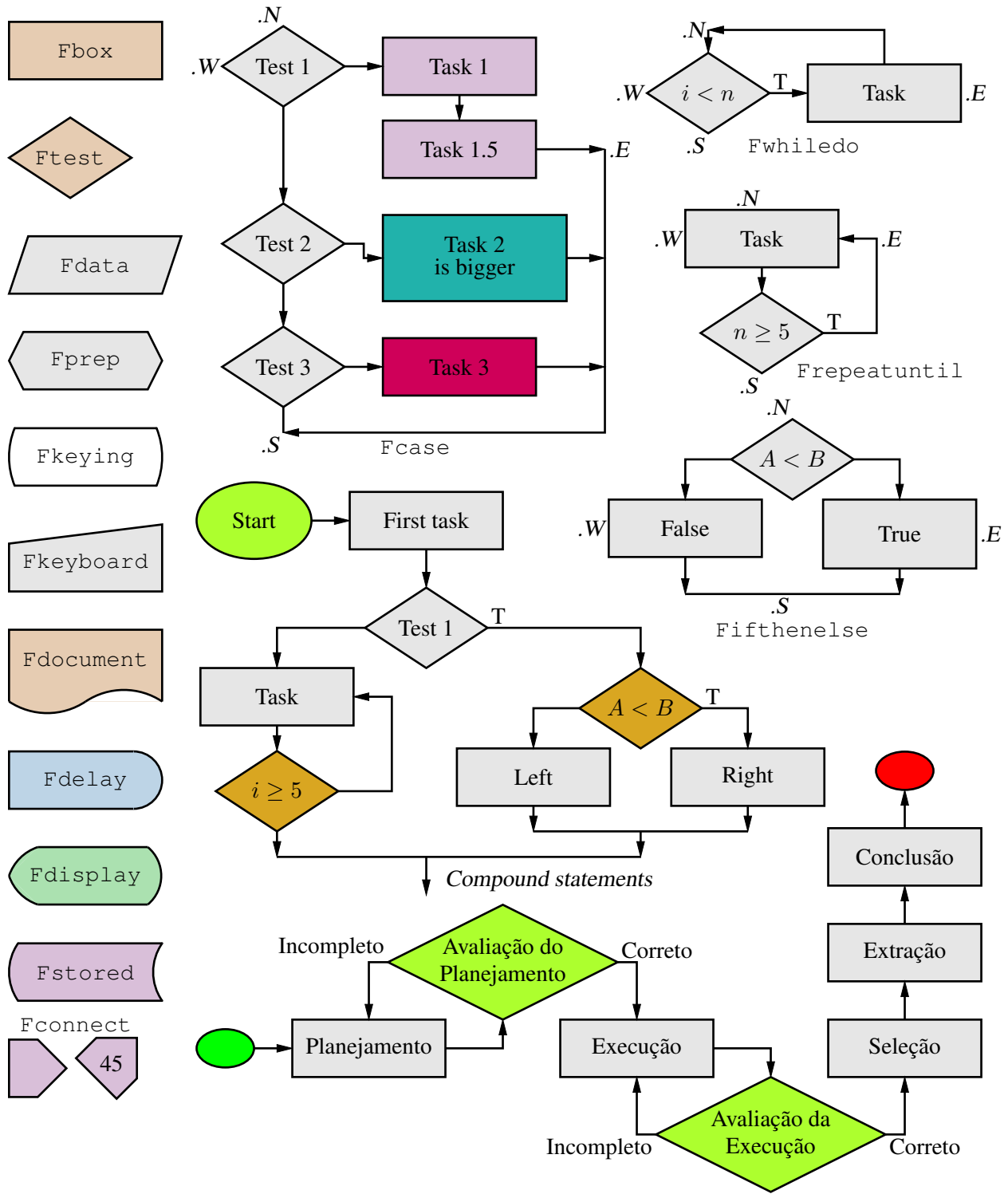


Figure 107: A flowchart sampler [Flow.m4].

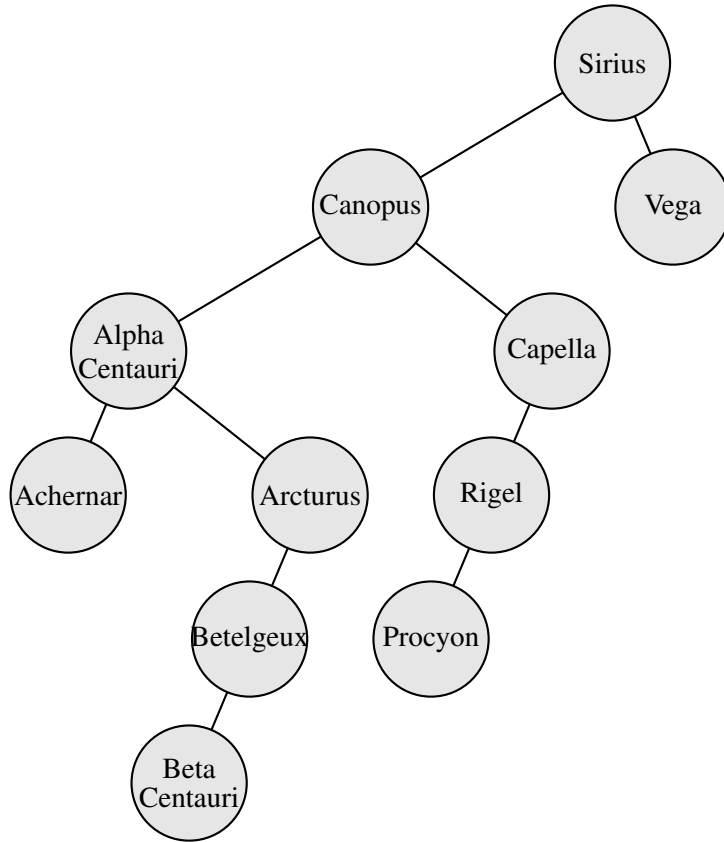


Figure 108: A binary tree [Btree.m4].

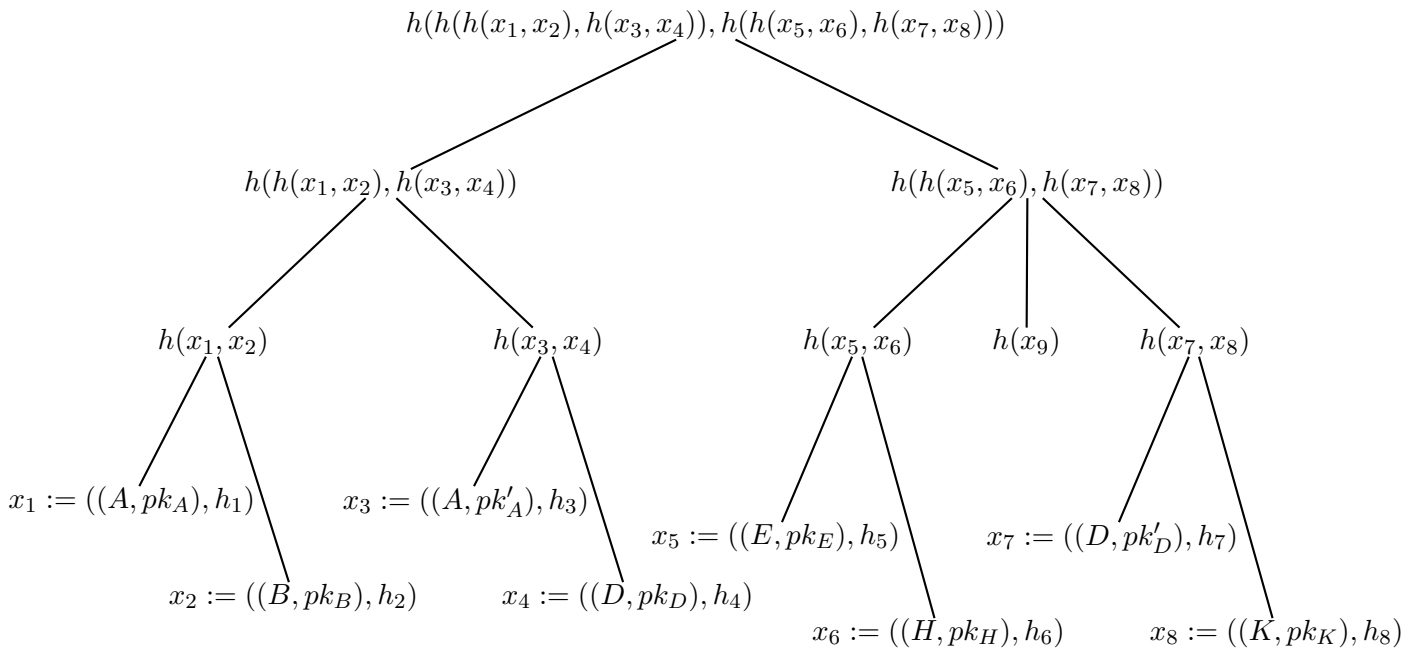


Figure 109: Another tree [Ttree.m4].



Figure 110: Overlaying a figure with line graphics [Incleps.m4].