TODO

Essential

- SSA is broken by simplify_loads() & branches rewriting/simplification
- add support for bitwise enums (wip)

Documentation

- document the API
- document the limitations of modifying ptrlists during list walking
- document the data structures
- document flow of data / architecture / code structure

Core

- if a variable has its address taken but in an unreachable BB then its MOD_ADDRESSABLE may be wrong and it won't be SSA converted.
  - let kill_insn() check killing of SYMADDR,
  - add the sym into a list and
  - recalculate the addressability before memops's SSA conversion
- bool_ctype should be split into internal 1-bit / external 8-bit

Testsuite

- there are 60 failing tests. They should be fixed (but most are non-trivial to fix).

Misc

- GCC's -Wenum-compare / clangs's -Wenum-conversion -Wassign-enum
- parse *__attribute__*((fallthrough))
- add support for format(printf()) (WIP by Ben Dooks)
- make use of UNDEFs (issues warnings, simplification, ... ?)
- make memory accesses more explicit: add EXPR_ACCESS (wip)
- it would be nice to do our own parsing of floating point (wip)
- some header files needed for crypto/ need __vector or __fp16
- some even need __complex

Optimization

- a lot of small simplifications are waiting to be upstreamed
- the domtree need to be rebuilt (or updated)
- critical edges need to be split

- the current way of doing CSE uses a lot of time
- add SSA based DCE
- add SSA based PRE
- Add SSA based SCCP
- add a pass to inline small functions during simplification.
- use better/more systematic use of internal verification framework
- tracking of operands size should be improved (WIP)
- OP_INLINE is sometimes in the way
- would be nice to strictly separate phases that don't changes the CFG and thus the dominance tree.

IR

- pseudos are untyped, it's usually OK but often it complicates things:
  - PSEUDO_REGs are defined by instructions and their type is normally retrievable via this defining instruction but in some cases they're not: for example, pseudos defined by ASM output.
  - PSEUDO_ARGs are considered as defined by OP_ENTRY and are used like this for liveness trackability but their type can't simply be retrieved via this instruction like PSEUDO_REGs are (with ->def->type).
  - PSEUDO_VALs are completely typeless.

  Maybe a few bits should be used to store some kind of low-level type.
- OP_SET should return a bool, always
- add IR instructions for va_arg() & friends
- add a possibility to import of file in "IR assembly"
- dump the symtable
- dump the CFG

LLVM

- fix ...

Internal backends

- it would be nice the upstream the code generator
- add a pass to transform 3-addresses code to 2-addresses
- add some basic register allocation
- add a pass to order the BBs and changes 2-ways CBR into one-way branches
- what can be done for x86?
- add support to add constraints in the MD rules

Longer term/to investigate

- attributes are represented as ctypes's alignment, modifiers & contexts but plenty of attributes doesn't fit, for example they need arguments.
  - format(printf, ...),
  - section("...")

    - assume_aligned(alignment[, offsert])
    - error("message"), warning("message")
    - ...
- should support "-Werror=..." ?
- All warning messages should include the option how to disable it. For example:

  "warning: Variable length array is used."

  should be something like:

  "warning: Variable length array is used. (-Wno-vla)"
- ptrlists must not have elements removed while being iterated; this should somehow be enforced.
- having 'struct symbol' used to represent symbols *and* types is quite handy but it also creates lots of problems and complications
- Possible mixup of symbol for a function designator being not a pointer? This seems to make evaluation of function pointers much more complex than needed.
- extend test-inspect to inspect more AST fields.
- extend test-inspect to inspect instructions.