File system in Stak Scheme

@raviqqe

June 23, 2024

Contents

- Stak Scheme
- File system in R7RS
- Implementation
- Future work

Stak Scheme

- A bytecode compiler and virtual machine (VM) for Scheme
- The compiler is written in Scheme.
- The VM is written in Rust.
- It aims to support R7RS-small.

File system in R7RS

Generic I/O

- Port type
  - `input-port?`, `output-port?`
  - `call-with-port`
- Read operations
  - `read-u8`, `read-string`, `read`
- Write operations
  - `write-u8`, `write-string`, `write`
- `close-port`

File system in R7RS

File operations

- `open-input-file`
- `open-output-file`
- `delete-file`
- `file-exists?`

Implementation

Generic I/O

- A `port` type
- vtable-ish implementation

```
(define-record-type port
  (make-port read write close)
  port?
  (read port-read)
  (write port-write)
  (close port-close))
```

Implementation

- Primitive file operations talks to libc directly.
- Rust's `std` crate doesn't expose some underlying details.
  - e.g. file descriptors

Opening files

```
(define (open-file path output)
  (let ((descriptor ($$open-file (string->path path) output)))
    (unless descriptor
      (error "cannot open file"))
    (make-port
      (lambda () ($$read-file descriptor))
      (lambda (byte) ($$write-file descriptor byte))
      (lambda () ($$close-file descriptor)))))
```

Future work

- More R7RS compatibility
- Efficient Scheme file compilation in Rust projects